第四章：人機介面行動技術-Arduino

南臺科技大學資訊工程系
李育強 助理教授

Outline

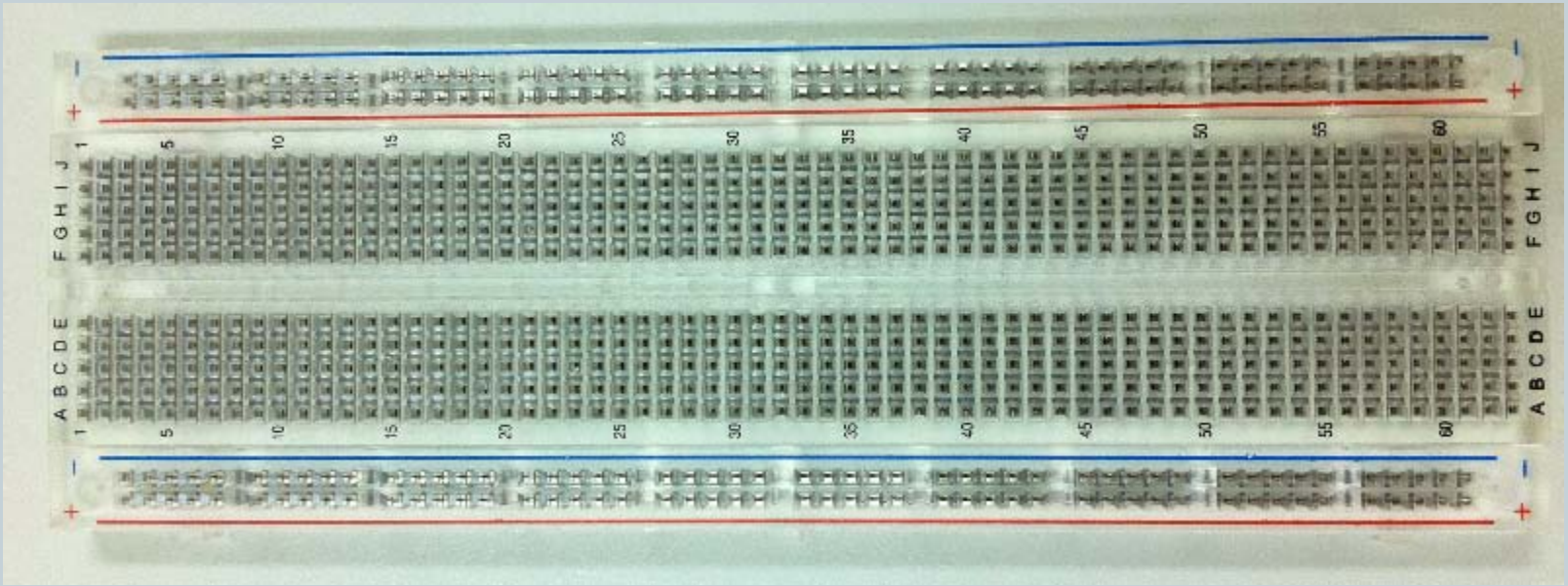
- 4-1 Arduino簡介
- 4-2 建立Arduino開發環境
- 4-3 利用Arduino 開發第一個程式
- **4-4 Arduino裝置人機互動設計**
- **4-5 Arduino與電腦端互動設計**
- **4-6 Arduino與行動裝置互動設計**

4-4 Arduino裝置人機互動設計

- 這節介紹的是Arduino裝置人機互動設計，我們將學習如何使用Arduino UNO搭配感測器元件(Sensor)來設計一些人機互動的實驗。
- 本節的目的如下：
  - 介紹並認識感測器元件。
  - 動手撰寫Arduino程式驅動感測器元件。
  - 動手接線瞭解簡單的電子電路。
  - 藉由簡單的實作得到更多的發想，期望能夠從實驗中組合出實用有趣的應用。

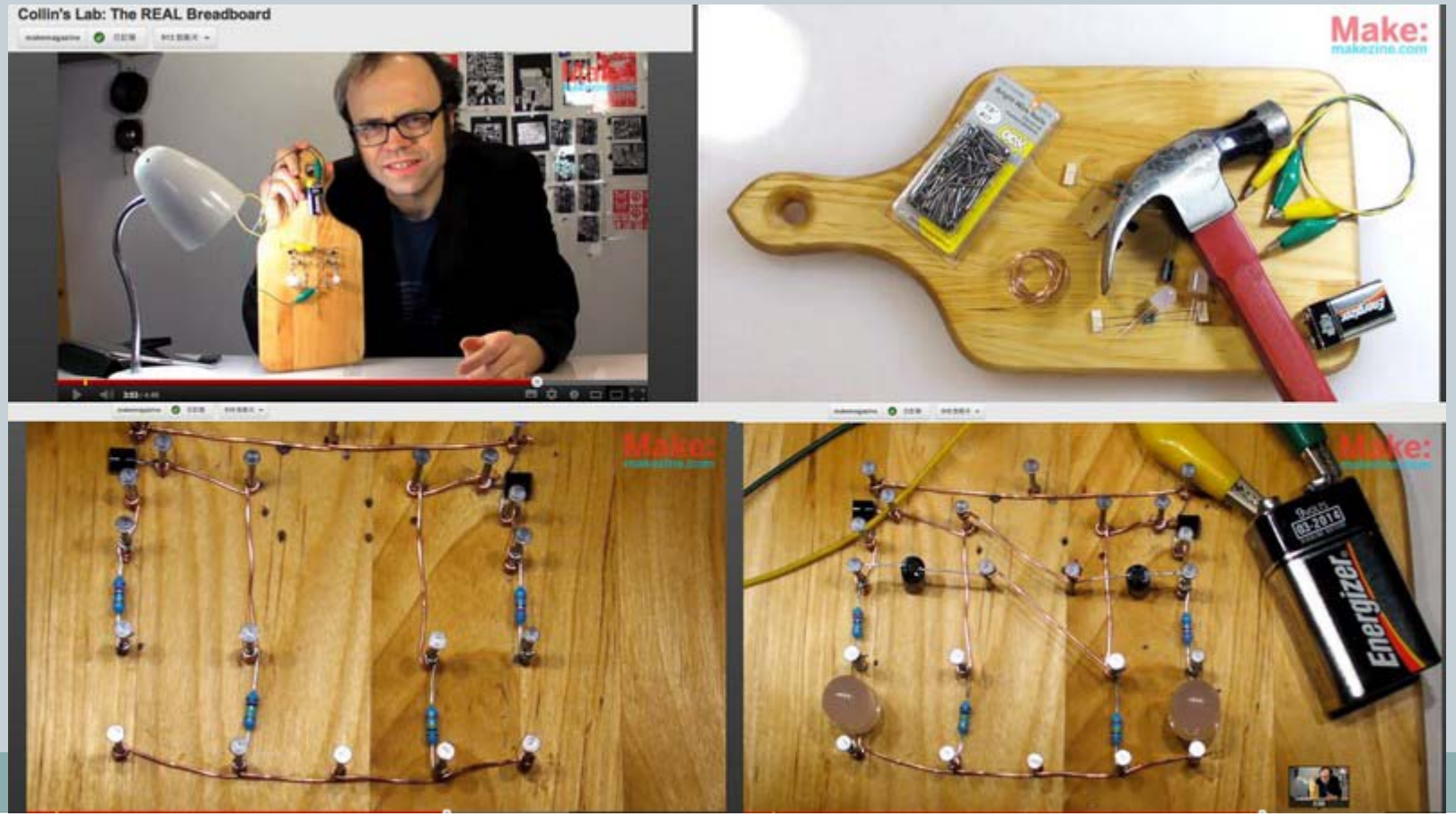
4-4 Arduino裝置人機互動設計

- 在實驗開始前，讓我們先介紹一下之後的實驗都會用到的工具「麵包板」的使用須知。



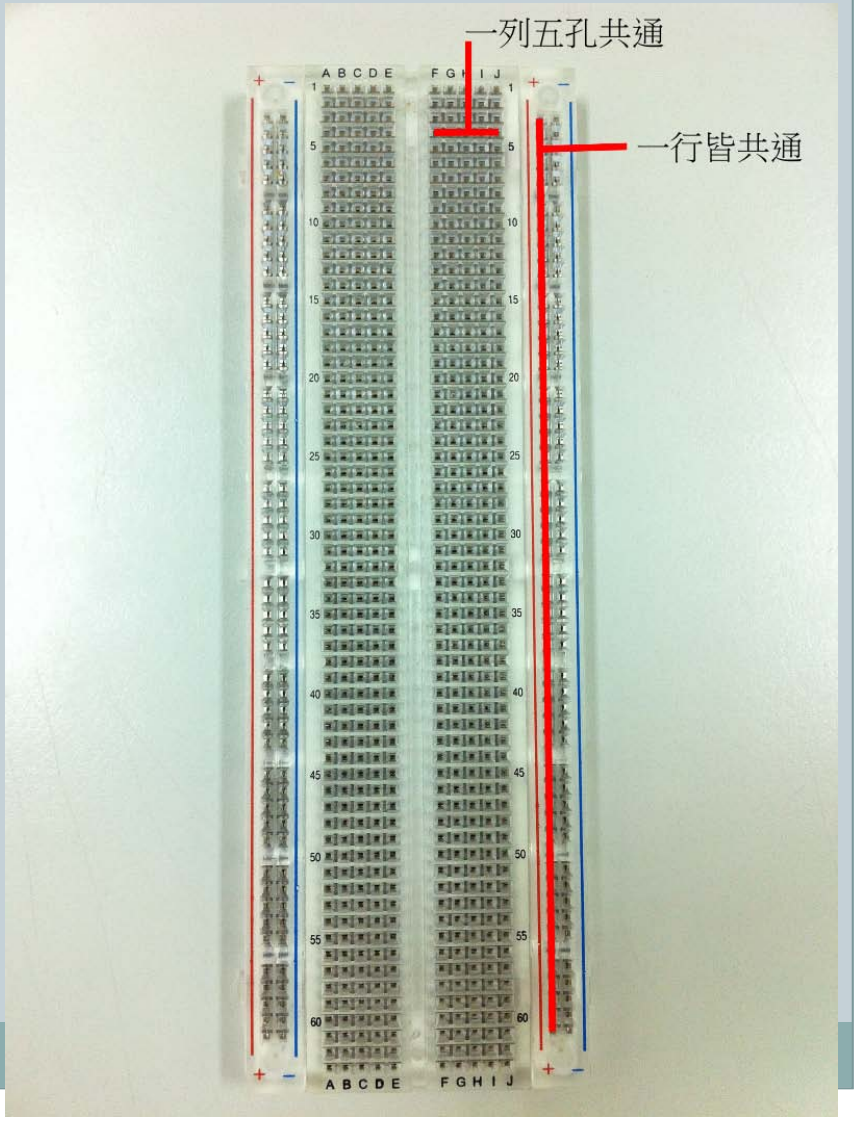
4-4 Arduino裝置人機互動設計

- 早期的電路都是接在麵包板上，隨著時代進步，麵包板進化到現在的模樣。



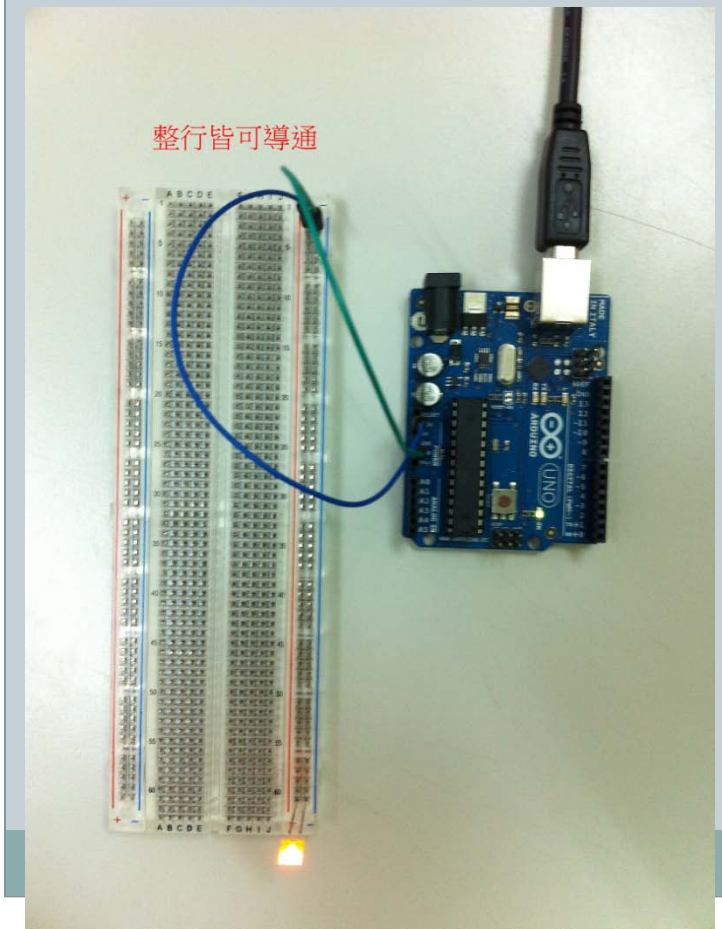
4-4 Arduino裝置人機互動設計

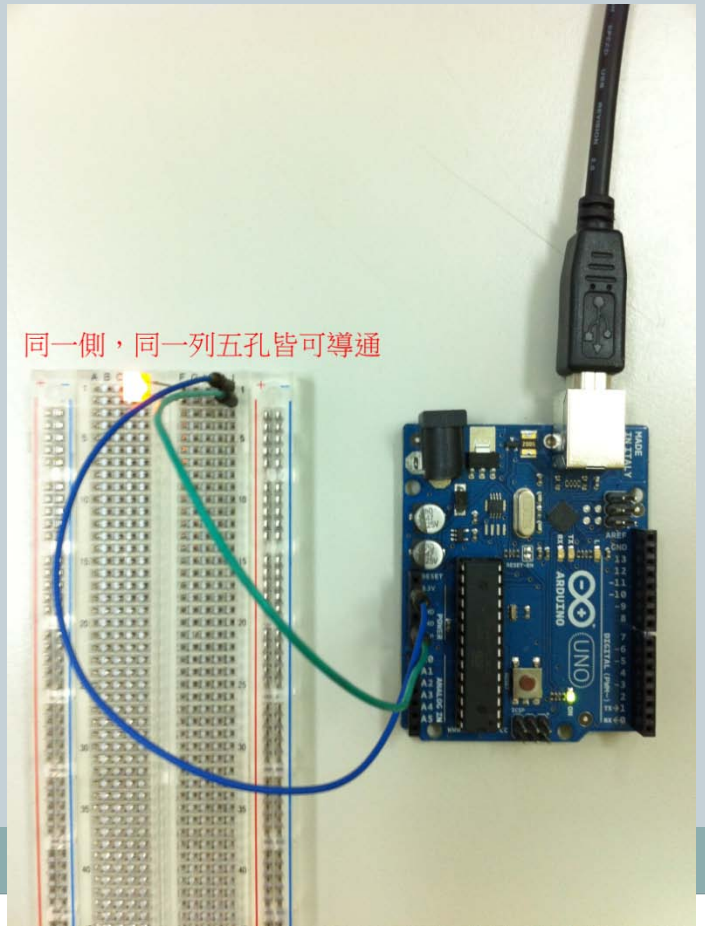
- 「麵包板」的使用須知：
  - 麵包板大致分為四到五行，左右兩側的插孔只要一孔有通電則整行都會導通。
  - 中間一列一列的部分則是一列五孔為一組，只要有一孔通電則其它四孔都會導通。



4-4 Arduino裝置人機互動設計

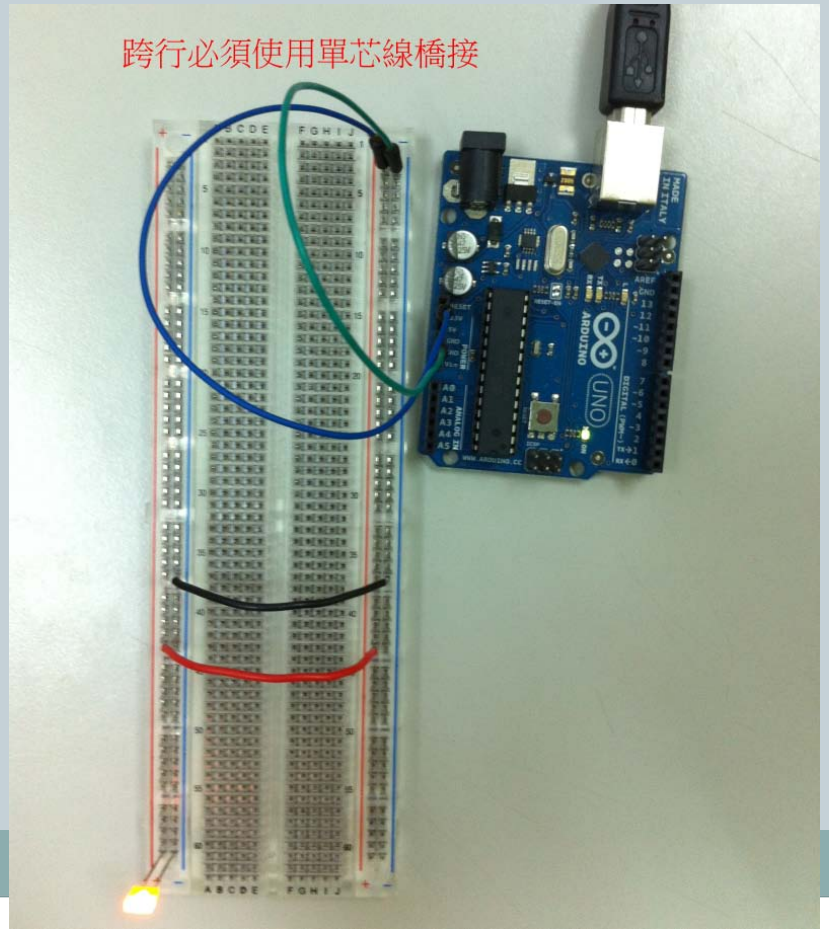
- 「麵包板」的使用須知：
  - 能夠導通的接法。下列是最一般的接法，不需要考慮橋接。

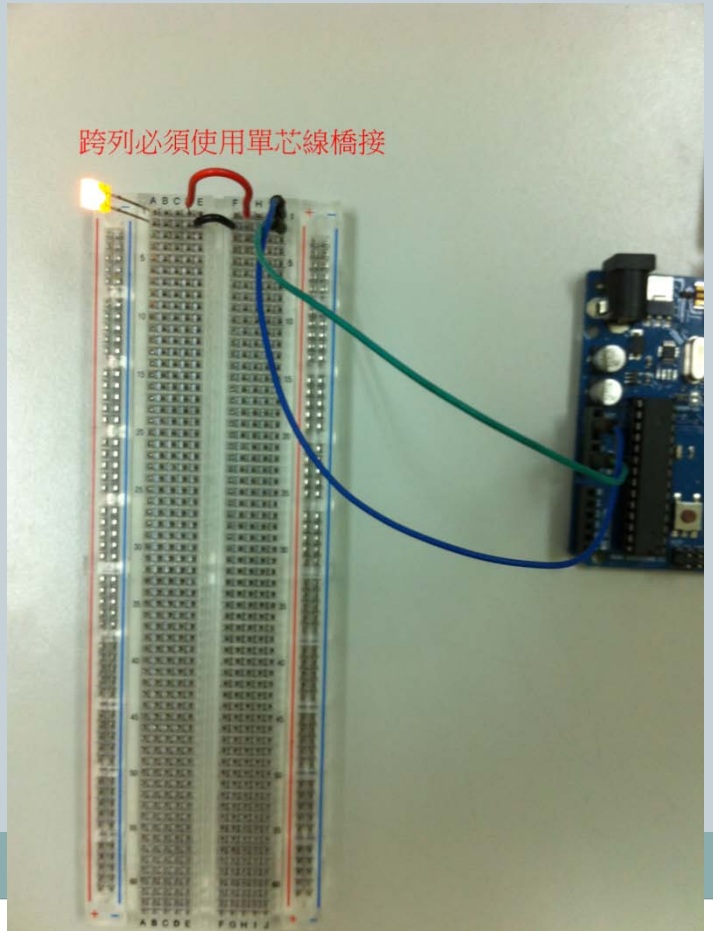




4-4 Arduino裝置人機互動設計

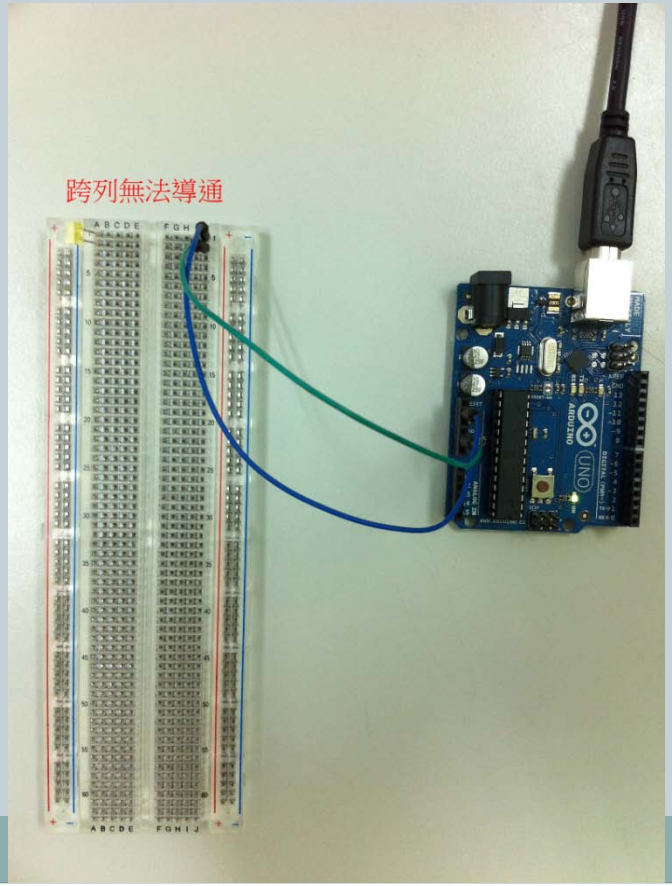
- 「麵包板」的使用須知：
  - 能夠導通的接法。如果需要跨行，可以使用單芯線橋接。





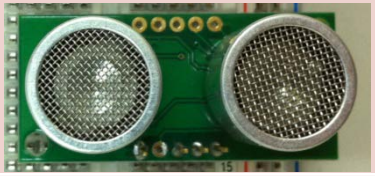
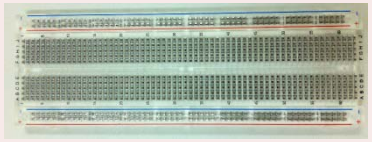
4-4 Arduino裝置人機互動設計

- 「麵包板」的使用須知：
  - 不能夠導通的接法。如果需要跨行，可以使用單芯線橋接。



4-4 Arduino裝置人機互動設計

- 實驗名稱：PRINT SONIC DISTANTCE。
  - 實驗目的：利用程式控制，蒐集超音波感測器元件所測得的距離。
  - 所需硬體材料：

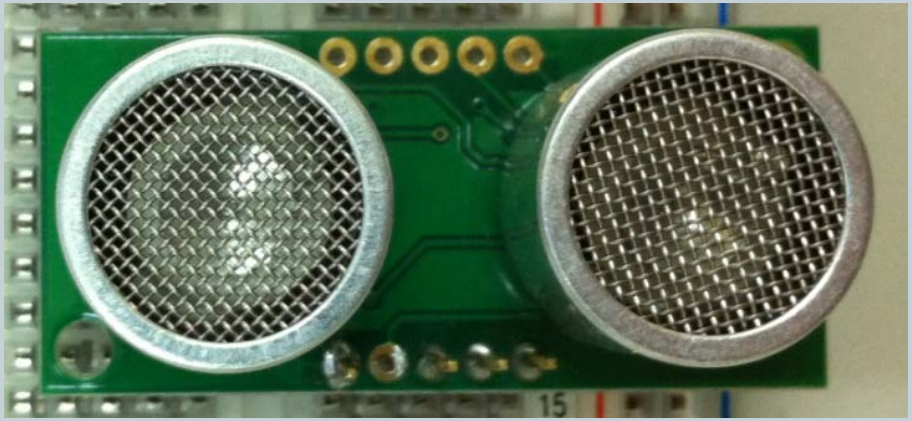
元件實體圖	數量	元件名稱
	1	超音波感測器元件
	1	麵包板
	1	Arduino UNO控制板
	4	單芯線

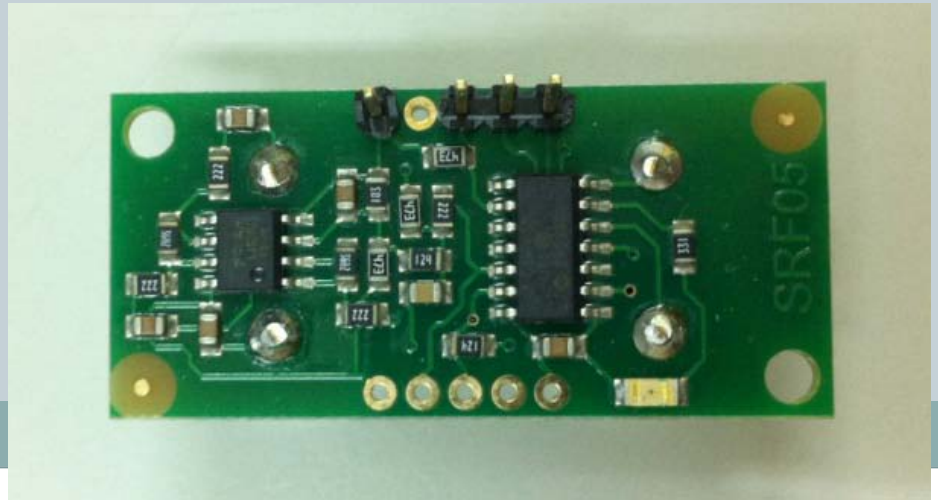
4-4 Arduino裝置人機互動設計

- 實驗名稱：PRINT SONIC DISTANTCE。
  - 感測器元件名稱：超音波感測器。
  - 感測器元件型號：SRF０５。
  - 感測器元件規格描述：
    - 這是一款超音波感測器，只要對它發送一個１０us的脈衝就會進行量測距離。
    - 超音波在空氣中傳播，途中碰到障礙物就立即返回來，超音波接收器收到反射波就立即停止計時。（超音波在空氣中的傳播速度為３４０公尺/秒）
    - 測得的數值只要除以５８就可以得到公分為單位的距離，或者也可以除以１４８就可以得到英吋為單位的距離。
    - 這款超音波感測器元件能夠測得的距離為３到４公尺，並且反應靈敏，能夠即時測得距離。
    - 我們可以利用超音波感測器元件製作倒車雷達之感測應用。

4-4 Arduino裝置人機互動設計

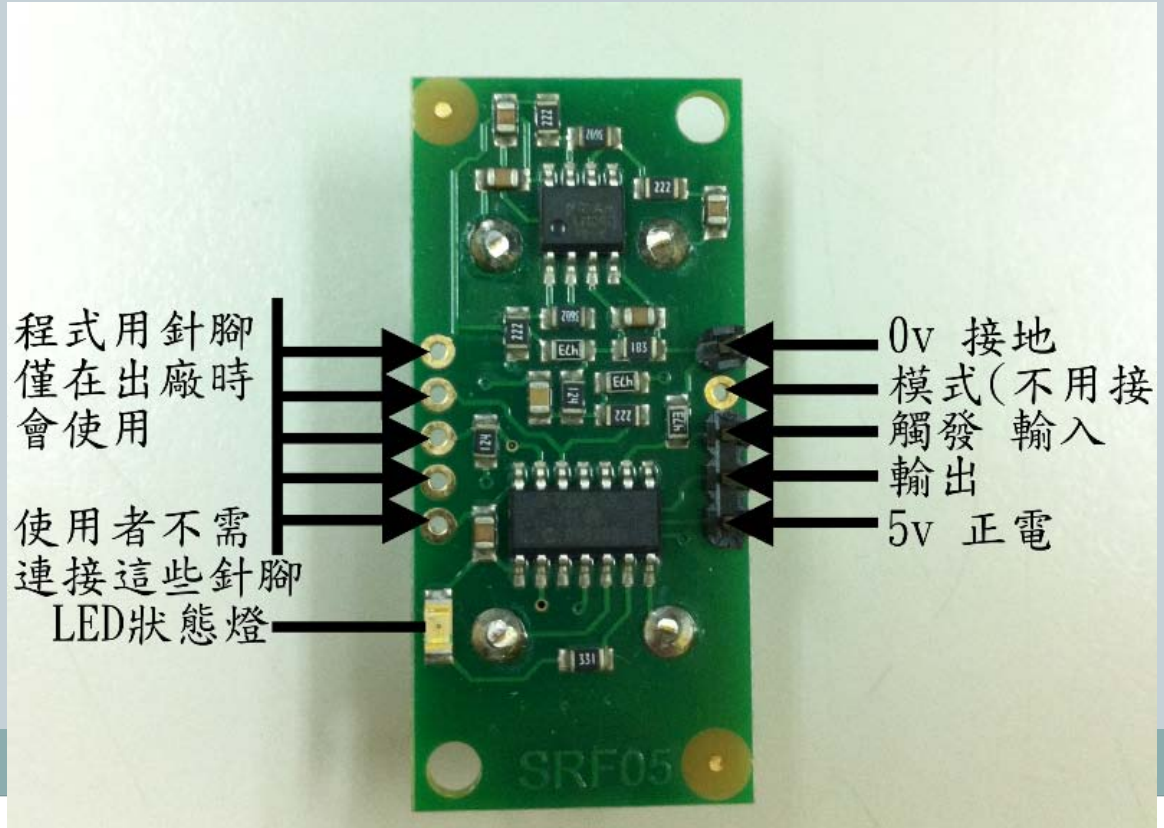
- 實驗名稱：PRINT SONIC DISTANTCE。
  - 下列為超音波感測器正背面之實體圖：





4-4 Arduino裝置人機互動設計

- 實驗名稱：PRINT SONIC DISTANTCE。
  - 超音波感測器之針腳定義圖：
  - 排列方向請依照印刷電路上的感測器元件型號所示。



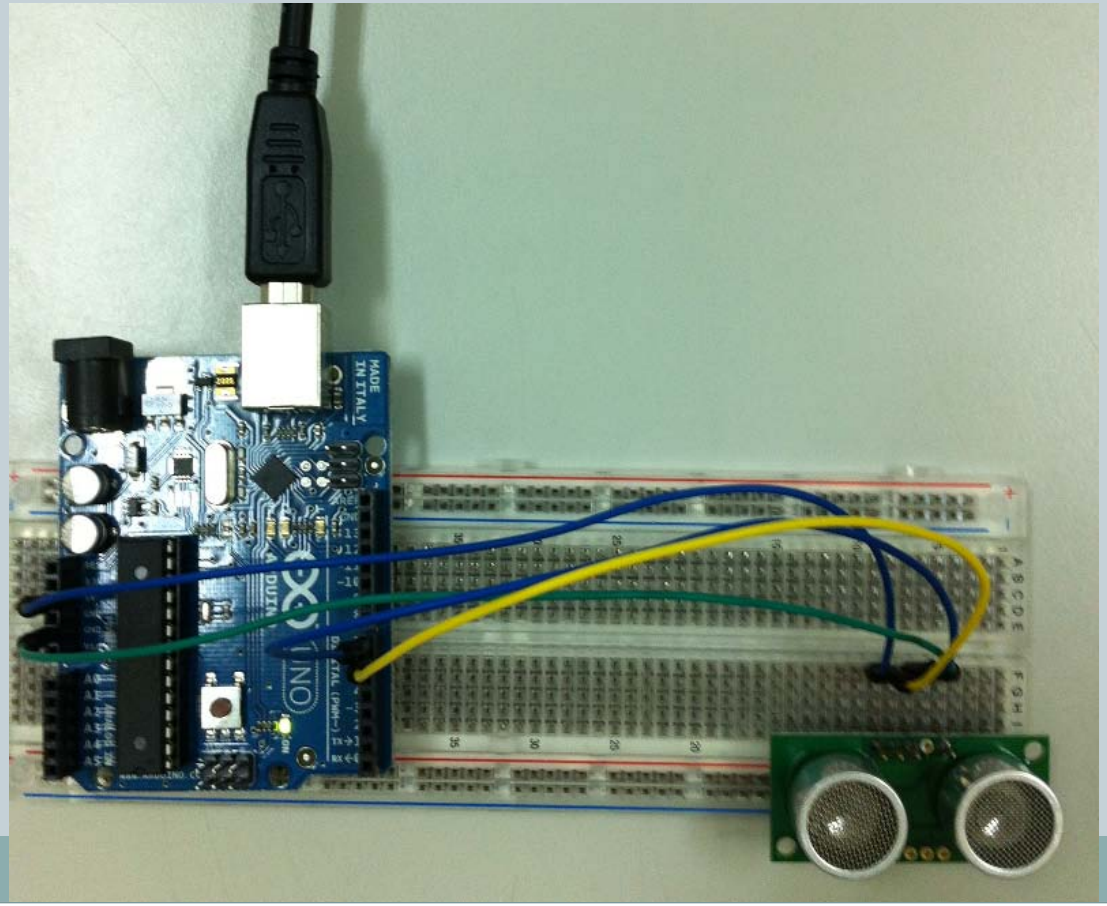
4-4 Arduino裝置人機互動設計

- 實驗名稱：PRINT SONIC DISTANTCE。
  - 程式碼：

```
#define ECHOPIN  6                              // 定義第六腳位為接收脈波
#define TRIGPIN 7                       //定義第七腳位為傳送脈波

void setup(){                           //Arduino程式的初始設定, 僅於上電時執行一次
 Serial.begin(9600);                    //設定傳輸率為9600
 pinMode(ECHOPIN, INPUT);       //指定接收脈波(第六腳位)為輸入
 pinMode(TRIGPIN, OUTPUT);      //指定傳送脈波(第七腳位)為輸出
}

void loop(){                            //Arduino程式的迴圈區段, 一直到斷電才結束動作
 digitalWrite(TRIGPIN, LOW);    //指定傳送脈衝(第七腳位)為低電位, 延遲2微秒
 delayMicroseconds(2);

 digitalWrite(TRIGPIN, HIGH);   // 傳送一個10微秒的脈波
 delayMicroseconds(10);

 digitalWrite(TRIGPIN, LOW);    //指定傳送脈波(第七腳位)為低電位
 int distance = pulseIn(ECHOPIN, HIGH);       //接收脈波(測得回傳的數值)
 distance= distance/58;                  //將測得的數值除以58得到單位為公分的距離
 Serial.println(distance);              //將測得的數值印出, 可由Arduino IDE的序列埠畫面看到數值
 delay(500);                            // 延遲0.5秒再進行一次測距離
}
```

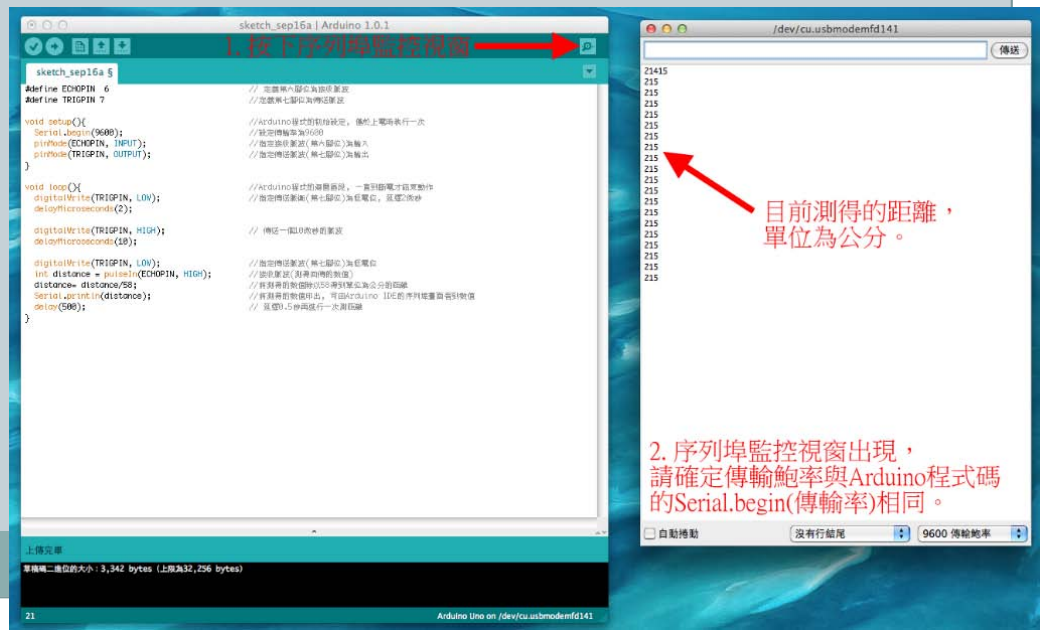
4-4 Arduino裝置人機互動設計

- 實驗名稱：PRINT SONIC DISTANTCE。
  - 接線圖：傳送脈波時，超音波感測器元件的LED狀態會閃爍。



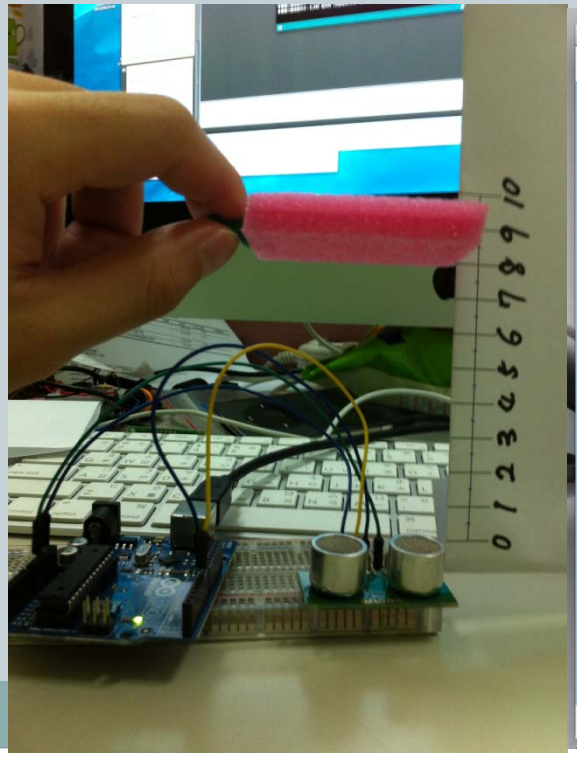
4-4 Arduino裝置人機互動設計

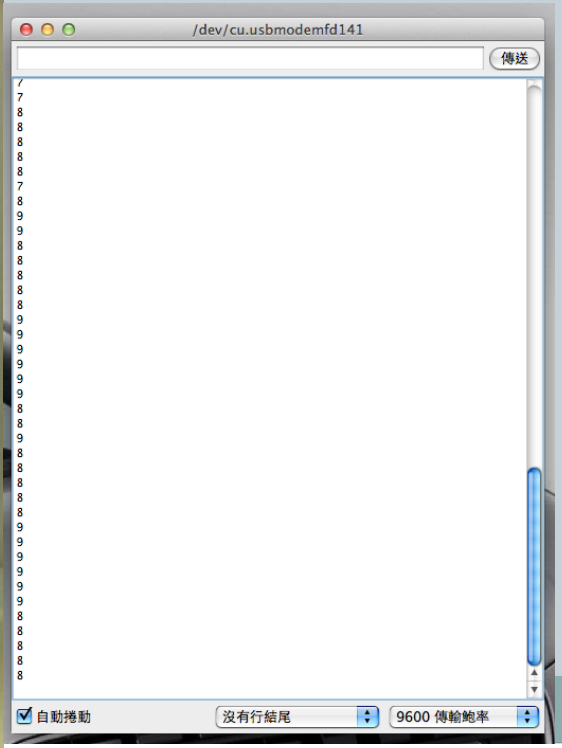
- 實驗名稱：PRINT SONIC DISTANTCE。
  - 說明：在完成接線以及程式撰寫，我們上傳程式至Arduino UNO無誤之後，按下Arduino IDE的「序列埠監控視窗」，可以看到目前測得的距離。畫面所呈現的數值２１５為放置感測器的桌面到天花板之間的距離，距離的單位是公分。
  - 另外，請確定傳輸鮑率與Arduino程式碼的Serial.begin(傳輸率)相同。



4-4 Arduino裝置人機互動設計

- 實驗名稱：PRINT SONIC DISTANTCE。
  - 說明：接著我們使用標好公分刻度的紙張作為對照圖，尺規的零點對齊超音波感測器元件的銀色外殼，並以一個障礙物阻擋來檢視所測得的結果，得到的數值與我們阻擋的刻度有所符合。





4-4 Arduino裝置人機互動設計

- 實驗名稱：PRINT SONIC DISTANTCE。
  - 實驗結果：
    - 在無法測得距離時會顯示數值為５２５，與我們一開始提到的測距範圍3-5公尺相符。
    - 而當太接近或是直接遮蓋住超音波感測器元件時也有可能會發生數值顯示為個位數甚至５２５的情形。
    - 若我們要將此元件應用至汽車倒車雷達的話，必須注意有可能會發生太近誤判的情形。

4-4 Arduino裝置人機互動設計

- 實驗名稱：MAGNETISM SWITCH。
  - 實驗目的：利用程式控制，得到磁力開關的狀態。
  - 所需硬體材料：

元件實體圖	數量	元件名稱
	1	磁力開關元件
	1	3 PIN杜邦線
	1	Arduino UNO控制板
	3	單芯線

4-4 Arduino裝置人機互動設計

- 實驗名稱：MAGNETISM SWITCH。
  - 感測器元件名稱：磁力開關元件。
  - 感測器元件型號：Magnetism。
  - 感測器元件規格描述：
    - 這是一款利用接近磁力則閉合，而遠離磁力則開啟的簡易磁力開關感測器元件。
    - 我們可以利用磁力開關元件做到門窗是否有緊閉或是檢測物體是否具有磁力的感測應用。

4-4 Arduino裝置人機互動設計

- 實驗名稱：MAGNETISM SWITCH。
  - 下列為磁力開關元件之實體圖：

4-4 Arduino裝置人機互動設計

- 實驗名稱：MAGNETISM SWITCH。
  - 磁力開關元件之針腳定義圖：
  - 排列方向請依照下圖所示。

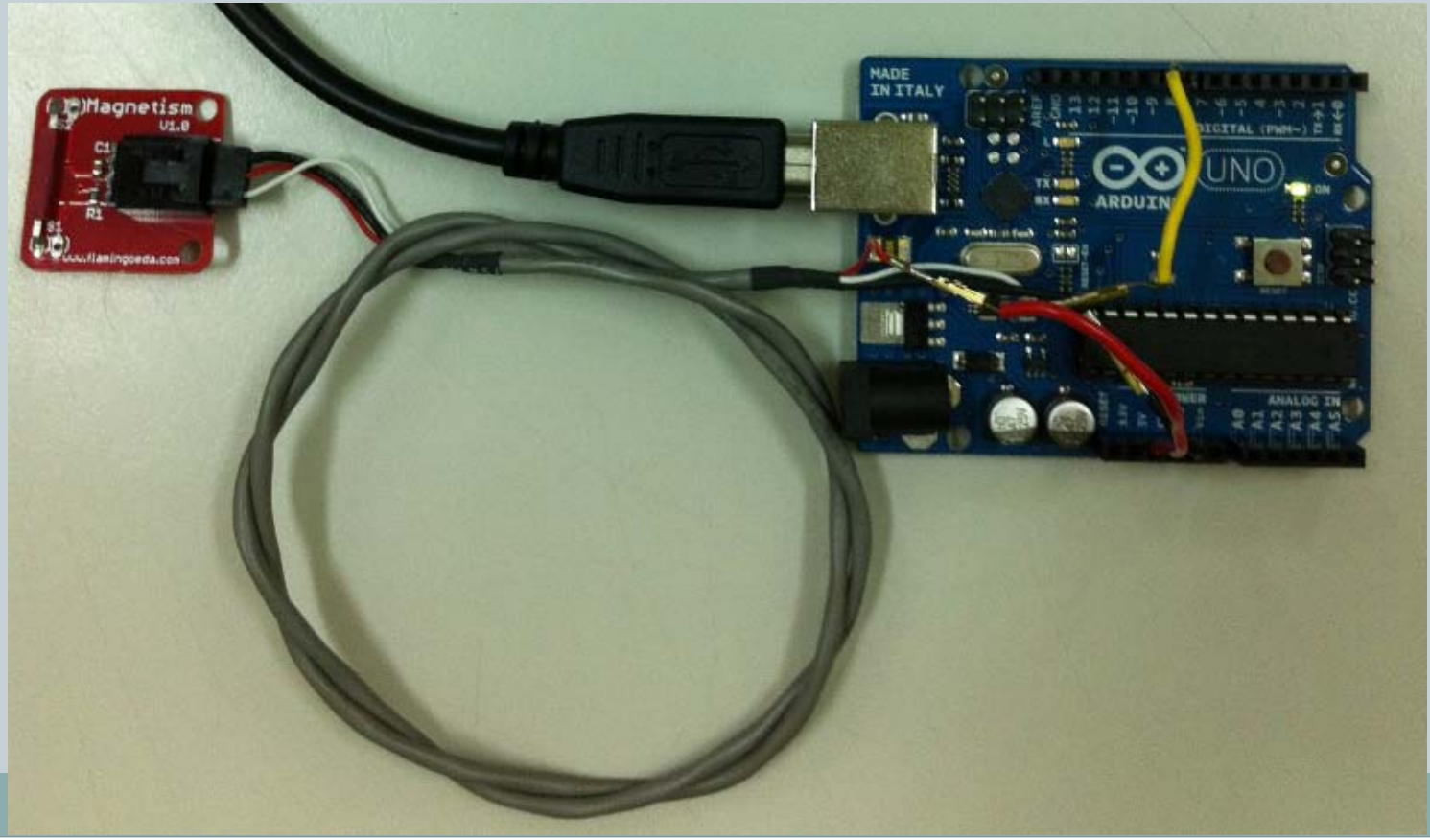
4-4 Arduino裝置人機互動設計

- 實驗名稱：PRINT SONIC DISTANTCE。
  - 程式碼：

```
#define MAGNETISM 8                          //定義第八腳位為磁力開關輸出

int magnetismState = 0;                                  //定義磁力開關的狀態為低電位

void setup(){                                //Arduino程式的初始設定, 僅於上電時執行一次
 Serial.begin(9600);                         //設定傳輸率為9600
 pinMode(MAGNETISM, INPUT);                  //指定接收脈波(第八腳位)為輸入
}

void loop(){                                 //Arduino程式的迴圈區段, 一直到斷電才結束動作
 magnetismState = digitalRead(MAGNETISM);    //將磁力開關輸出的狀態存入magnetismState

 if (magnetismState == HIGH)                 //當磁力開關被磁性物質所吸引時則會轉為高電位
 {
  Serial.println("ON");                      //印出ON, 可以在序列埠監控視窗看到
 }
 else                                        //當磁力開關沒有被磁性物質所吸引時則會轉為低電位
 {
  Serial.println("OFF");                     //印出OFF, 可以在序列埠監控視窗看到
 }
 delay(500);                                 // 延遲0.5秒再進行一次感測
}
```

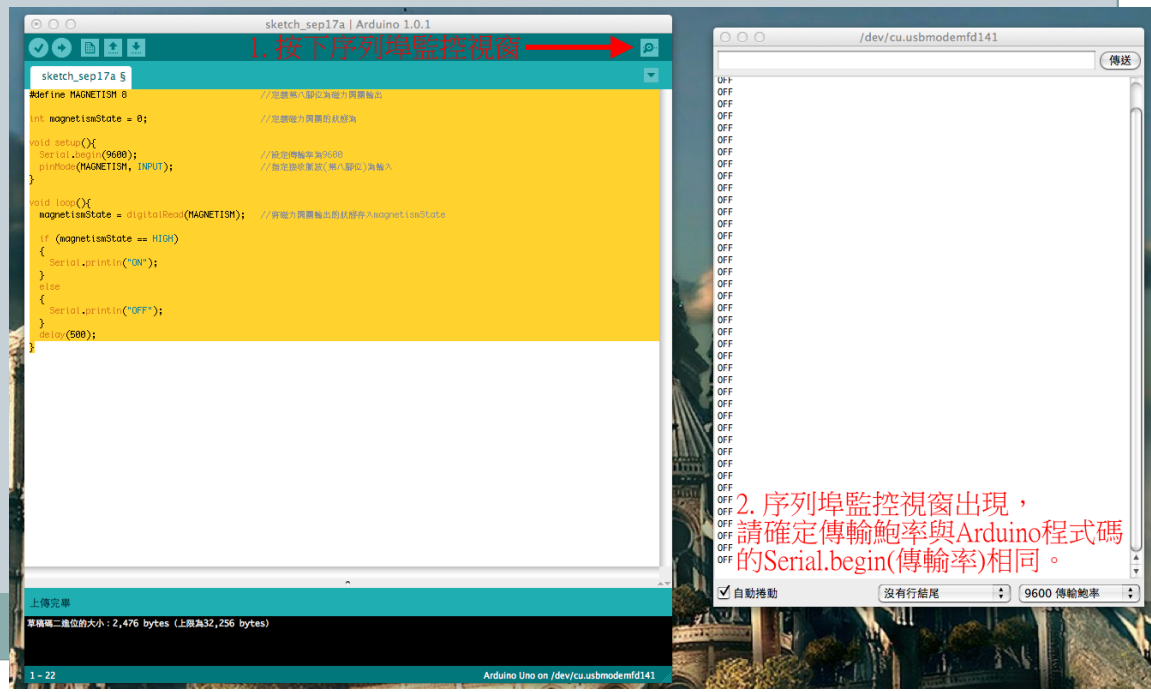
4-4 Arduino裝置人機互動設計

- 實驗名稱：MAGNETISM SWITCH。
  - 接線圖：



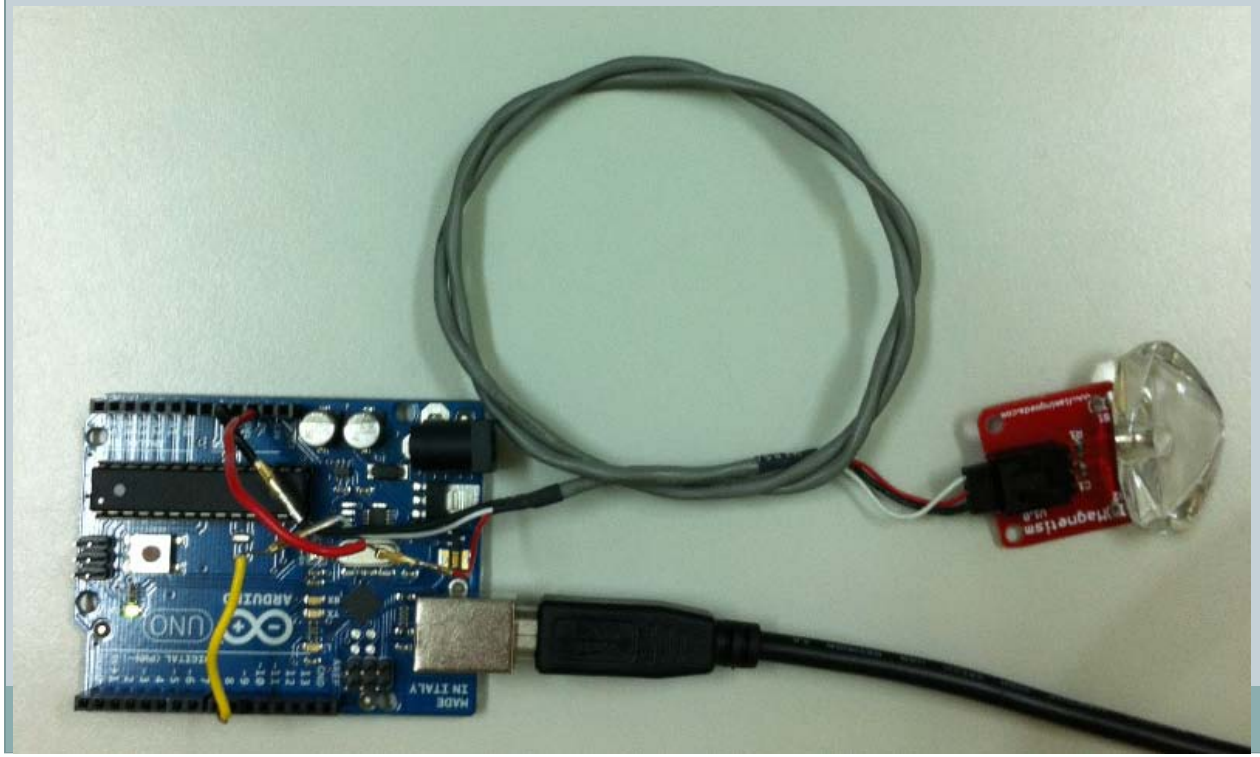
4-4 Arduino裝置人機互動設計

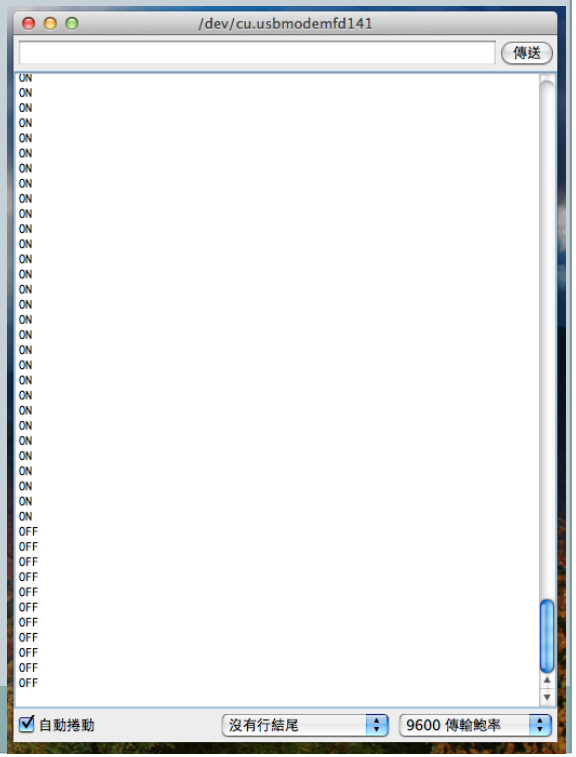
- 實驗名稱：MAGNETISM SWITCH。
  - 說明：在完成接線以及程式撰寫，我們上傳程式至Arduino UNO無誤之後，按下Arduino IDE的「序列埠監控視窗」，可以看到目前磁力開關元件的狀態。
  - 另外，請確定傳輸鮑率與Arduino程式碼的Serial.begin(傳輸率)相同。



4-4 Arduino裝置人機互動設計

- 實驗名稱：MAGNETISM SWITCH。
  - 說明：接著我們使用磁鐵接觸磁力開關元件的感測點，可以發現序列埠監控視窗中的狀態由OFF變為ON，當磁鐵離開磁力開關元件時，可以發現序列埠監控視窗中的狀態由ON變為OFF。





4-4 Arduino裝置人機互動設計

- 實驗名稱：MAGNETISM SWITCH。
  - 實驗結果：
    - 當接近磁力時，磁力開關則閉合。
    - 遠離磁力時，磁力開關則開啟。
    - 可以做到門窗是否有緊閉或是檢測物體是否具有磁力的感測應用。

4-4 Arduino裝置人機互動設計

- 實驗名稱：PASSIVE INFRA-RED SENSOR。
  - 實驗目的：利用程式控制，得到紅外線感測器元件的狀態。
  - 所需硬體材料：

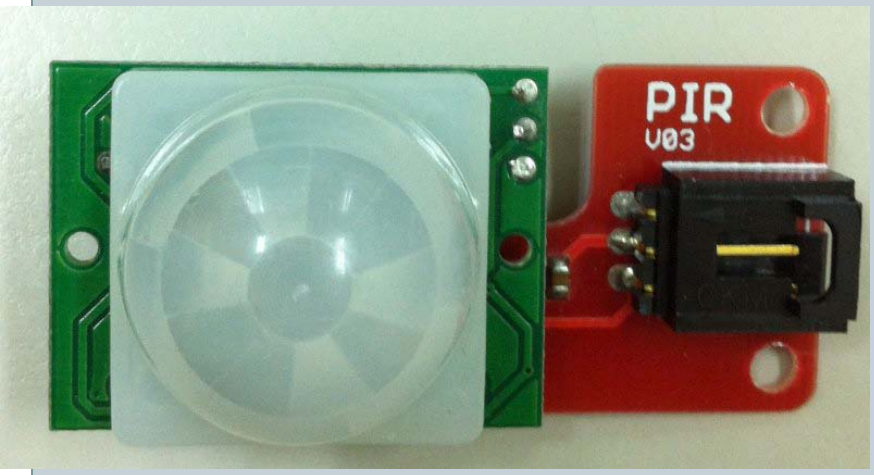
元件實體圖	數量	元件名稱
	1	紅外線感測器元件
	1	3 PIN杜邦線
	1	Arduino UNO控制板
	3	單芯線

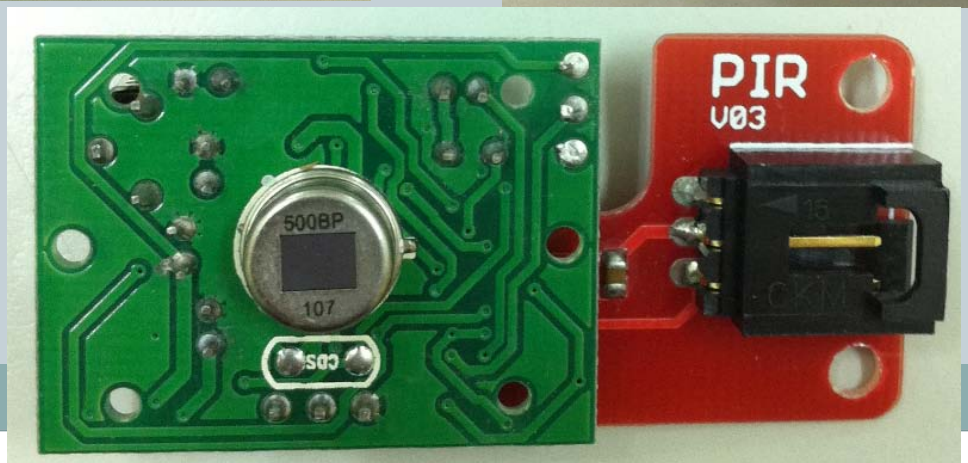
4-4 Arduino裝置人機互動設計

- 實驗名稱：PASSIVE INFRA-RED SENSOR。
  - 感測器元件名稱：紅外線感測器元件。
  - 感測器元件型號：PIR V０３。
  - 感測器元件規格描述：
    - 這是一款能夠感測環境的移動物體所發出的紅外線（熱能）變化的感測器元件。
    - 當紅外線感測器元件感測到移動物體所發出的紅外線時，它會輸出一個高電位訊號，而我們可以根據這個反應做相關應用。
    - 紅外線感測器元件的輸入電壓為直流３.３伏特至５伏特。
    - 紅外線感測器元件的感測移動物體的有效距離為６公尺以內。
    - 我們可以利用紅外線感測器元件做到居家安全警報感測或是省電照明應用。

4-4 Arduino裝置人機互動設計

- 實驗名稱：PASSIVE INFRA-RED SENSOR。
  - 下列為紅外線感測器元件之實體圖：





4-4 Arduino裝置人機互動設計

- 實驗名稱：PASSIVE INFRA-RED SENSOR。
  - 紅外線感測器元件之針腳定義圖：
  - 排列方向請依照下圖所示。

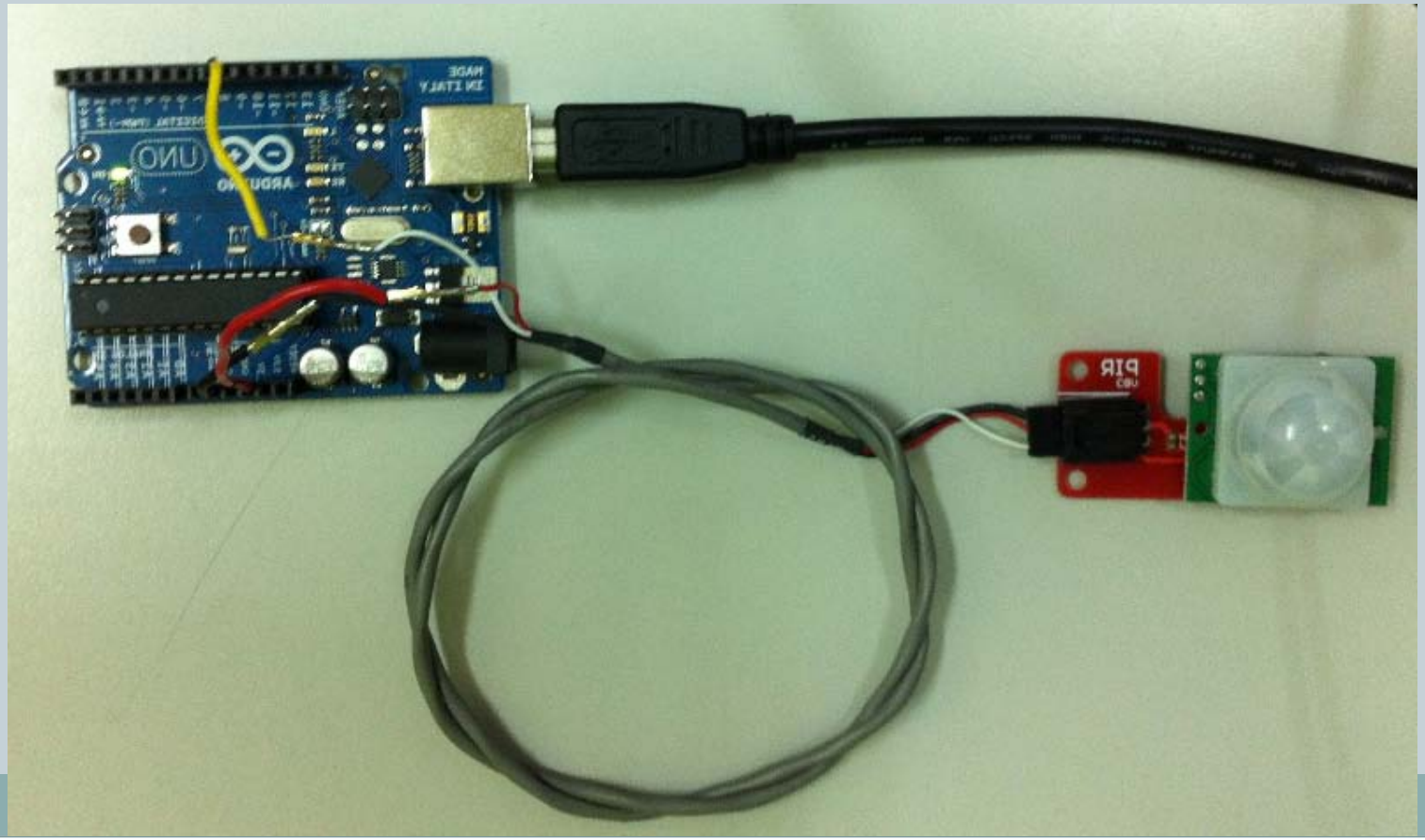
4-4 Arduino裝置人機互動設計

- 實驗名稱：PASSIVE INFRA-RED SENSOR。
  - 程式碼：

```
#define PIR 8                                   //定義第八腳位為紅外線感測器元件輸出

int pirState = 0;                               //定義紅外線感測的狀態為低電位

void setup(){                                   //Arduino程式的初始設定, 僅於上電時執行一次
 Serial.begin(9600);                            //設定傳輸率為9600
 pinMode(PIR , INPUT);                          //指定紅外線感測器元件輸出(第八腳位)為輸入
}

void loop(){                                    //Arduino程式的迴圈區段, 一直到斷電才結束動作
 pirState = digitalRead(PIR);                   //將紅外線感測器元件輸出的狀態存入pirState

  if (pirState == HIGH)                         //當紅外線感測器元件感測到物體經過時則會轉為高電位
 {
  Serial.println("ON");                         //印出ON, 可以在序列埠監控視窗看到
 }
 else                                           //當沒有物體經過紅外線感測器元件時則會轉為低電位
 {
  Serial.println("OFF");                        //印出OFF, 可以在序列埠監控視窗看到
 }
 delay(500);                                    // 延遲0.5秒再進行一次感測
}
```

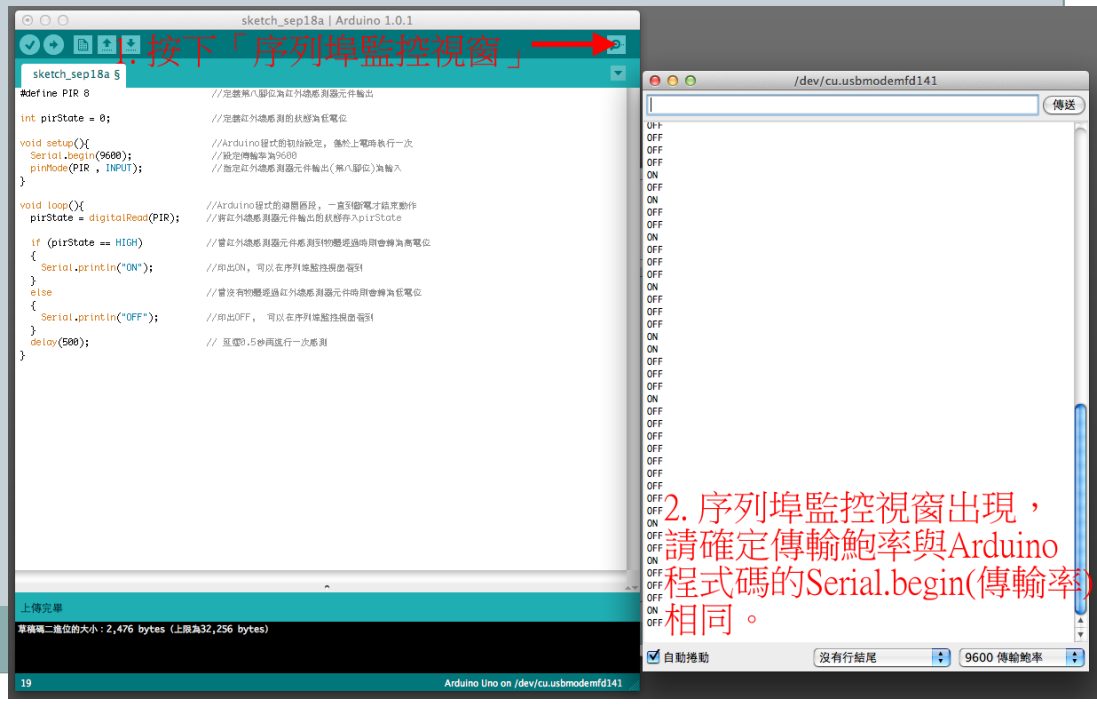
4-4 Arduino裝置人機互動設計

- 實驗名稱：PASSIVE INFRA-RED SENSOR。
  - 接線圖：



4-4 Arduino裝置人機互動設計

- 實驗名稱：PASSIVE INFRA-RED SENSOR。
  - 說明：在完成接線以及程式撰寫，我們上傳程式至Arduino UNO無誤之後，按下Arduino IDE的「序列埠監控視窗」，可以看到目前紅外線感測器元件的狀態。
  - 另外，請確定傳輸鮑率與Arduino程式碼的Serial.begin(傳輸率)相同。



4-4 Arduino裝置人機互動設計

- 實驗名稱：PASSIVE INFRA-RED SENSOR。
  - 說明：程式執行後會，序列埠監控視窗會持續列出OFF，當我們以手掌拂過紅外線感測器元件時會發現序列埠監控視窗中會出現ON，但是卻無法持續顯示為ON，這時候可以調整紅外線感測器元件背面的旋鈕來延長ON的時間，也就可以達到像洗手間那樣，只要發現有人進入就會點亮日光燈，並且持續點亮，一直到紅外線感應器元件內建的計時器的計時時間到才會熄滅日光燈。

4-4 Arduino裝置人機互動設計

- 實驗名稱：PASSIVE INFRA-RED SENSOR。
  - 實驗結果：
    - 藉由實驗我們可以瞭解紅外線感測元件的使用其實很容易，程式碼與之前介紹的磁力開關元件之程式碼非常相似，而接線圖也與磁力開關元件非常相似，但兩者的差異在於紅外線感測器元件能夠感測到的距離是比磁力開關元件還要遠，能夠應用的層面就不太相似，沒有重複到的問題。
    - 從實驗中我們發現，正常時候，序列埠監控視窗顯示為ON的時間約為０.３秒，如果我們將計時器的調整旋鈕順時針調整到最底，可以得到大約２０秒的延遲時間，若是延時不足夠的話還可以再搭配Arduino來撰寫延時程式，就可以可以做到居家安全警報感測或是省電照明應用。

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD。
  - 實驗目的：
    - 在Arduino UNO上建立一個會持續發出序列埠訊息的序列埠程式，我們暫時先不討論在Arduino UNO上安裝感測器元件。
    - 接下來我們會在電腦端設計一個應用程式來接收Arduino UNO所發出的序列埠訊息。
  - 所需開發軟體：
    - Microsoft Visual Studio 2010 C#。
    - Arduino IDE。

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD。
  - 所需硬體材料：

元件實體圖	元件數量	元件名稱
	1	Arduino UNO控制板

4 - 5 Arduino與電腦端互動設計

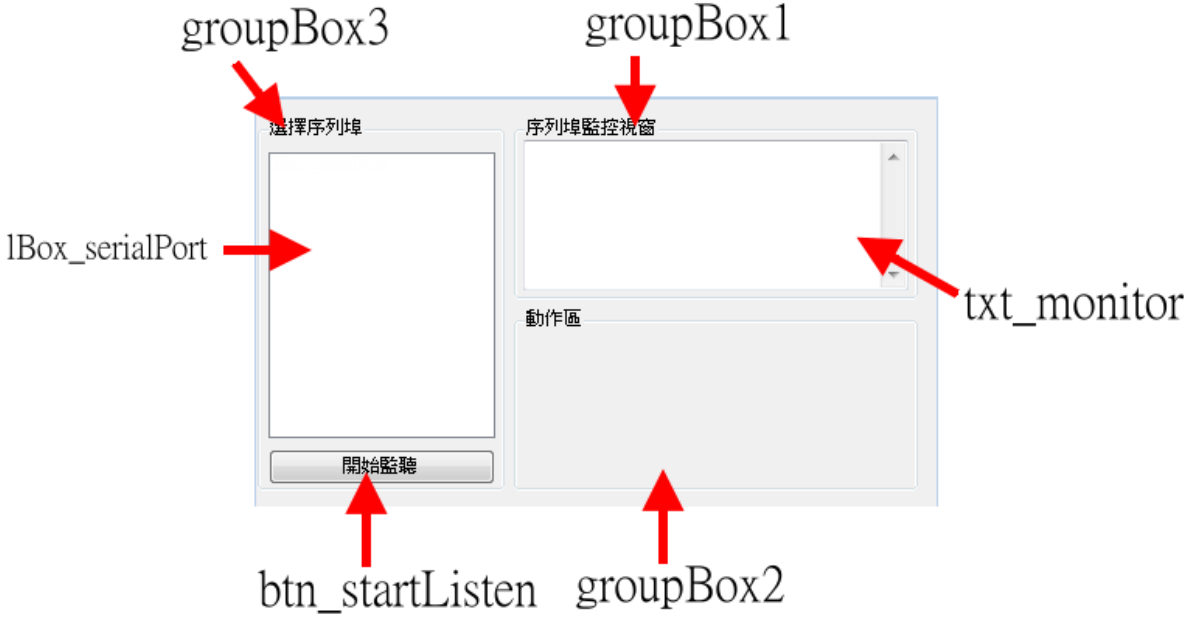
- 實驗名稱：HELLO WORLD。
  - 實驗目的：
    - 在Arduino UNO上建立一個會持續發出序列埠訊息的序列埠程式，我們暫時先不討論在Arduino UNO上安裝感測器元件。
    - 接下來我們會在電腦端設計一個應用程式來接收Arduino UNO所發出的序列埠訊息。
  - 所需開發軟體：
    - Microsoft Visual Studio 2010 C#
    - Arduino IDE

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD。
  - Microsoft Visual Studio 2010 C#部分：
    - 執行Microsoft Visual Studio 2010我們按下「新增專案」的按鈕（或者是按下鍵盤的「Control + Shift + N」）來新增一個「Visual C#」的「Windows Form應用程式」。
    - 應用程式的框架版本可以選擇「.NET Framework 4」即可。最後就是指定這個專案的「名稱」、「位置」以及「方案名稱」，請依照自己的需求來命名。
    - 在我們選擇好「.NET Framework 4」並且指定專案的「名稱」、「位置」以及「方案名稱」之後，我們就可以按下「確定」就可以建立一個「Windows Form應用程式」了。
    - 建立好專案後，我們需要先建立「Windows Form應用程式」的介面。

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD。
  - 建立「Windows Form應用程式」的介面，介面編排如下列圖示：



4-5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD。
  - 建立「Windows Form應用程式」的介面，介面編排說明如下列所示：
    - Form１ 大小：４５０px * ３００px。
    - Form１ 標題：SERIALPORT MONITOR 。
    - Form１ 事件：Form1_FormClosing、Form1_Load 。
    - Form１ 取消最大化。
    - btn_startListen 大小：１４６px * ２３px。
    - btn_startListen 標題：開始監聽。
    - btn_startListen 事件： btn_startListen_Click。
    - btn_startListen 座標：７px * ２１３px 。
    - lBox_serialPort 大小：１４６px * １８４px。
    - lBox_serialPort 座標：７px * ２２px 。

4-5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD。
  - 建立「Windows Form應用程式」的介面，介面編排說明如下列所示：
    - groupBox１ 大小：２６０px * １１７px。
    - groupBox１ 標題：序列埠監控視窗。
    - groupBox１ 座標：１６６px * １２px 。
    - groupBox２ 大小：２６０px * １１７px。
    - groupBox２ 標題：動作區。
    - groupBox２ 座標：１６６px * １３５px 。
    - groupBox３ 大小：１５９px * ２４０px。
    - groupBox３ 標題：選擇序列埠。
    - groupBox３ 座標：１px * １２px 。

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD。
  - 建立「Windows Form應用程式」，「Form1.CS」的程式碼。
    - using System;
    - using System.Collections.Generic;
    - using System.ComponentModel;
    - using System.Data;
    - using System.Drawing;
    - using System.Linq;
    - using System.Text;
    - using System.Windows.Forms;
    - using System.IO.Ports; //必須引用System.IO.Ports才可以直接找到SerialPort。

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD。
  - 建立「Windows Form應用程式」，「Form1.CS」的程式碼。

```
namespace CS_WFAPP_SERIALPORT_MONITOR
{
    public partial class Form1 : Form
    {
        //定義一個序列埠元件。
        protected SerialPort serialPort = new SerialPort();
       //定義一個計時器元件。
        protected Timer timer = new Timer();
public Form1()
        {
            InitializeComponent();
        }
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD。
  - 建立「Windows Form應用程式」，「Form1.CS」的程式碼。

```
//Form產生後的進入點事件。
private void Form1_Load(object sender, EventArgs e)
{
        //將系統目前連接的序列埠列表交給lBox_serialPort作為資料來源列表出來。
        lBox_serialPort.DataSource = SerialPort.GetPortNames();
        //設定計時器元件的觸發間隔時間，１０００毫秒等於１秒
        timer.Interval = 1000;
        //設定計時器元件的觸發事件
        timer.Tick += new EventHandler(timer_Tick);
}
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD。
  - 建立「Windows Form應用程式」，「Form1.CS」的程式碼。

```
//計時器元件的觸發事件
private void timer_Tick(object sender, EventArgs e) {
        try
        {
            //將序列埠元件所讀取到的資料行再加上一個換行符號，然後交給txt_monitor顯示。
            txt_monitor.Text += serialPort.ReadLine() + System.Environment.NewLine;
        }
        catch { }
    }
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD。
  - 建立「Windows Form應用程式」，「Form1.CS」的程式碼。

```
//Form關閉時的事件
        private void Form1_FormClosing(object sender, FormClosingEventArgs e)
        {
            //判斷序列埠連線狀態是否為連線中
            if (serialPort.IsOpen)
            {
                //關閉連線
                serialPort.Close();
            }
            //釋放序列埠元件
            serialPort.Dispose();
        }
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD。
  - 建立「Windows Form應用程式」，「Form1.CS」的程式碼。

```
//按下開始監聽/停止監聽按鈕的事件。
private void btn_startListen_Click(object sender, EventArgs e)
{
  //當按鈕的標題為停止監聽時。
  if (btn_startListen.Text == "停止監聽")
  {
    //當序列埠列表致能被取消時。
    if (lBox_serialPort.Enabled == false)
    {
      //將序列埠列表設為致能。
      lBox_serialPort.Enabled = true;
      //將groupBox1的標題回復成「序列埠監控視窗」。
      groupBox1.Text = "序列埠監控視窗";

      try
      {
        //判斷序列埠連線狀態是否為連線中。
        if (serialPort.IsOpen)
        {
          //關閉連線。
          serialPort.Close();
        }
      }
      catch
      { }

      //停止計時器元件。
      timer.Enabled = false;
      //將btn_startListen標題回復為「開始監聽」。
      btn_startListen.Text = "開始監聽";
    }
  }
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD。
  - 建立「Windows Form應用程式」，「Form1.CS」的程式碼。

```
//當按鈕狀態為開始監聽時。
        else
        {
          //當序列埠列表有選擇一個項目時。
          if (lBox_serialPort.SelectedIndex > -1)
          {
            //將序列埠列表所選的項目作為序列埠元件監聽的對象。
            serialPort.PortName = lBox_serialPort.SelectedValue.ToString();
            //設定序列埠元件的傳輸鮑率為９６００，要跟Arduino的傳輸鮑率一樣。
            serialPort.BaudRate = 9600;
            //將groupBox1的標題設定為「序列埠監控視窗 - 連接埠名稱」提示使用者目前所選的連接埠是什麼。
            groupBox1.Text = string.Format("序列埠監控視窗 - {0}", lBox_serialPort.SelectedValue.ToString());
            //將txt_monitor的內容設定為「序列埠監控視窗 - 連接埠名稱」再加上一個換行符號，提示使用者目前所選的連接埠是什麼。
            txt_monitor.Text = string.Format("選擇序列埠：{0}，開始監控。{1}", lBox_serialPort.SelectedValue.ToString(), System.Environment.NewLine);
            //取消序列埠列表的致能。
            lBox_serialPort.Enabled = false;

            try
            {
              //判斷序列埠連線狀態是否為關閉中。
              if (!serialPort.IsOpen)
              {
                //開啟連線。
                serialPort.Open();
              }
            }
            catch
            { }

            //開啟計時器元件。
            timer.Enabled = true;
            //將btn_startListen標題設定為「停止監聽」。
            btn_startListen.Text = "停止監聽";
          }
        }
    }

      }
    }
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」，「Form1.Designer.cs」的程式碼。

```
namespace CS_WFAPP_SERIALPORT_MONITOR
{
  partial class Form1
  {
    /// <summary>
    /// 設計工具所需的變數。
    /// </summary>
    private System.ComponentModel.IContainer components = null;

    /// <summary>
    /// 清除任何使用中的資源。
    /// </summary>
    /// <param name="disposing">如果應該處置 Managed 資源則為 true，否則為 false。</param>
    protected override void Dispose(bool disposing)
    {
      if (disposing && (components != null))
      {
        components.Dispose();
      }
      base.Dispose(disposing);
    }

    #region Windows Form 設計工具產生的程式碼
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」，「Form1.Designer.cs」的程式碼。

```
/// <summary>
        /// 此為設計工具支援所需的方法 - 請勿使用程式碼編輯器
        /// 修改這個方法的內容。
        /// </summary>
        private void InitializeComponent()
        {
            this.txt_monitor = new System.Windows.Forms.TextBox();
            this.groupBox1 = new System.Windows.Forms.GroupBox();
            this.groupBox2 = new System.Windows.Forms.GroupBox();
            this.groupBox3 = new System.Windows.Forms.GroupBox();
            this.btn_startListen = new System.Windows.Forms.Button();
            this.lBox_serialPort = new System.Windows.Forms.ListBox();
            this.groupBox1.SuspendLayout();
            this.groupBox2.SuspendLayout();
            this.groupBox3.SuspendLayout();
            this.SuspendLayout();
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」，「Form1.Designer.cs」的程式碼。

```
//
// txt_monitor
//
this.txt_monitor.Location = new System.Drawing.Point(6, 14);
this.txt_monitor.Multiline = true;
this.txt_monitor.Name = "txt_monitor";
this.txt_monitor.ScrollBars = System.Windows.Forms.ScrollBars.Both;
this.txt_monitor.Size = new System.Drawing.Size(248, 97);
this.txt_monitor.TabIndex = 0;
//
// groupBox1
//
this.groupBox1.Controls.Add(this.txt_monitor);
this.groupBox1.Location = new System.Drawing.Point(166, 12);
this.groupBox1.Name = "groupBox1";
this.groupBox1.Size = new System.Drawing.Size(260, 117);
this.groupBox1.TabIndex = 1;
this.groupBox1.TabStop = false;
this.groupBox1.Text = "序列埠監控視窗";
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」，「Form1.Designer.cs」的程式碼。

```
//
// groupBox2
//
this.groupBox2.Controls.Add(this.btn_ligthsOff);
this.groupBox2.Controls.Add(this.btn_lightsUp);
this.groupBox2.Controls.Add(this.btn_sendMessage);
this.groupBox2.Location = new System.Drawing.Point(166, 135);
this.groupBox2.Name = "groupBox2";
this.groupBox2.Size = new System.Drawing.Size(260, 117);
this.groupBox2.TabIndex = 2;
this.groupBox2.TabStop = false;
this.groupBox2.Text = "動作區";
this.groupBox2. Enabled = false;
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」，「Form1.Designer.cs」的程式碼。

```
            //
            // groupBox3
            //
            this.groupBox3.Controls.Add(this.btn_startListen);
            this.groupBox3.Controls.Add(this.lBox_serialPort);
            this.groupBox3.Location = new System.Drawing.Point(1, 12);
            this.groupBox3.Name = "groupBox3";
            this.groupBox3.Size = new System.Drawing.Size(159, 240);
            this.groupBox3.TabIndex = 3;
            this.groupBox3.TabStop = false;
            this.groupBox3.Text = "選擇序列埠";
            //
            // btn_startListen
            //
            this.btn_startListen.Location = new System.Drawing.Point(7, 213);
            this.btn_startListen.Name = "btn_startListen";
            this.btn_startListen.Size = new System.Drawing.Size(146, 23);
            this.btn_startListen.TabIndex = 1;
            this.btn_startListen.Text = "開始監聽";
            this.btn_startListen.UseVisualStyleBackColor = true;
            this.btn_startListen.Click += new System.EventHandler(this.btn_startListen_Click);
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」，「Form1.Designer.cs」的程式碼。

```
            //
            // lBox_serialPort
            //
            this.lBox_serialPort.FormattingEnabled = true;
            this.lBox_serialPort.ItemHeight = 12;
            this.lBox_serialPort.Location = new System.Drawing.Point(7,
22);
            this.lBox_serialPort.Name = "lBox_serialPort";
            this.lBox_serialPort.Size = new System.Drawing.Size(146, 184);
            this.lBox_serialPort.TabIndex = 0;
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」，「Form1.Designer.cs」的程式碼。

```
            //
            // Form1
            //
            this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 12F);
            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
            this.ClientSize = new System.Drawing.Size(434, 262);
            this.Controls.Add(this.groupBox3);
            this.Controls.Add(this.groupBox2);
            this.Controls.Add(this.groupBox1);
            this.MaximizeBox = false;
            this.Name = "Form1";
            this.Text = "SERIALPORT MONITOR";
            this.FormClosing += new System.Windows.Forms.FormClosingEventHandler(this.Form1_FormClosing);
            this.Load += new System.EventHandler(this.Form1_Load);
            this.groupBox1.ResumeLayout(false);
            this.groupBox1.PerformLayout();
            this.groupBox2.ResumeLayout(false);
            this.groupBox3.ResumeLayout(false);
            this.ResumeLayout(false);

        }

        #endregion
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」，「Form1.Designer.cs」的程式碼。

```
private System.Windows.Forms.TextBox txt_monitor;
        private System.Windows.Forms.GroupBox groupBox1;
        private System.Windows.Forms.GroupBox groupBox2;
        private System.Windows.Forms.GroupBox groupBox3;
        private System.Windows.Forms.ListBox lBox_serialPort;
        private System.Windows.Forms.Button btn_startListen;
    }
}
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD。
  - 建立「Arduino」的程式碼：

```
//Arduino程式的初始設定, 僅於上電時執行一次
void setup(){
//設定傳輸率為9600
Serial.begin(9600);
}

//Arduino程式的迴圈區段, 一直到斷電才結束動作
void loop(){
//印出Hello World!, 可以在序列埠監控視窗看到
Serial.println(“Hello World!”);
// 延遲1秒再進行一次動作
 delay(1000);
}
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD。
  - 動作說明：
    - 「Arduino」撰寫完成後請將程式「上傳」至Arduino UNO。
    - 「Windows Form應用程式」介面以及程式碼撰寫完成之後，請將程式「編譯」，沒有錯誤之後就可以「開始偵錯」。
    - 在「Windows Form應用程式」介面中選擇Arduino UNO所在的序列埠，然後按下「開始監聽」，可以看到序列埠監控視窗開始接收到訊息。
    - 訊息如下列所示：
      - 選擇序列埠：COM4，開始監控。
      - Hello World!
      - Hello World!
      - Hello World!
      - Hello World!
      - Hello World!
      - Hello World!
      - ...

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD。
  - 動作說明：
    - 觀察「Windows Form應用程式」介面中的「序列埠監控視窗」是否一秒接收到一次Hello World!的訊息，並且持續不中斷。
    - 觀察「Windows Form應用程式」介面中按下「開始監聽」後，其按鈕的標題是否轉為「停止監聽」，並且無法選擇序列埠。
    - 觀察「Windows Form應用程式」介面中按下「停止監聽」後，其按鈕的標題是否轉為「開始監聽」，並且可以選擇序列埠，而「序列埠監控視窗」是否停止監聽來自Arduino的訊息。

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD。
  - 實驗結果：
    - 經過這個實驗後，我們就能夠自行撰寫監聽序列埠的程式。
    - 我們可以監聽Arduino並且依照監聽到的訊息來做情境模擬。
    - 例如：讓Arduino裝上超音波感測元件，當超音波元件感測到有物體接近時可以發送訊一個序列埠訊息讓電腦端接收到，而電腦端就可以根據這個訊息來分析狀況。

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 所需硬體材料：

元件實體圖	元件數量	元件名稱
	1	ＬＥＤ燈（任何顏色皆可）
	1	Arduino UNO控制板

4 - 5 Arduino與電腦端互動設計

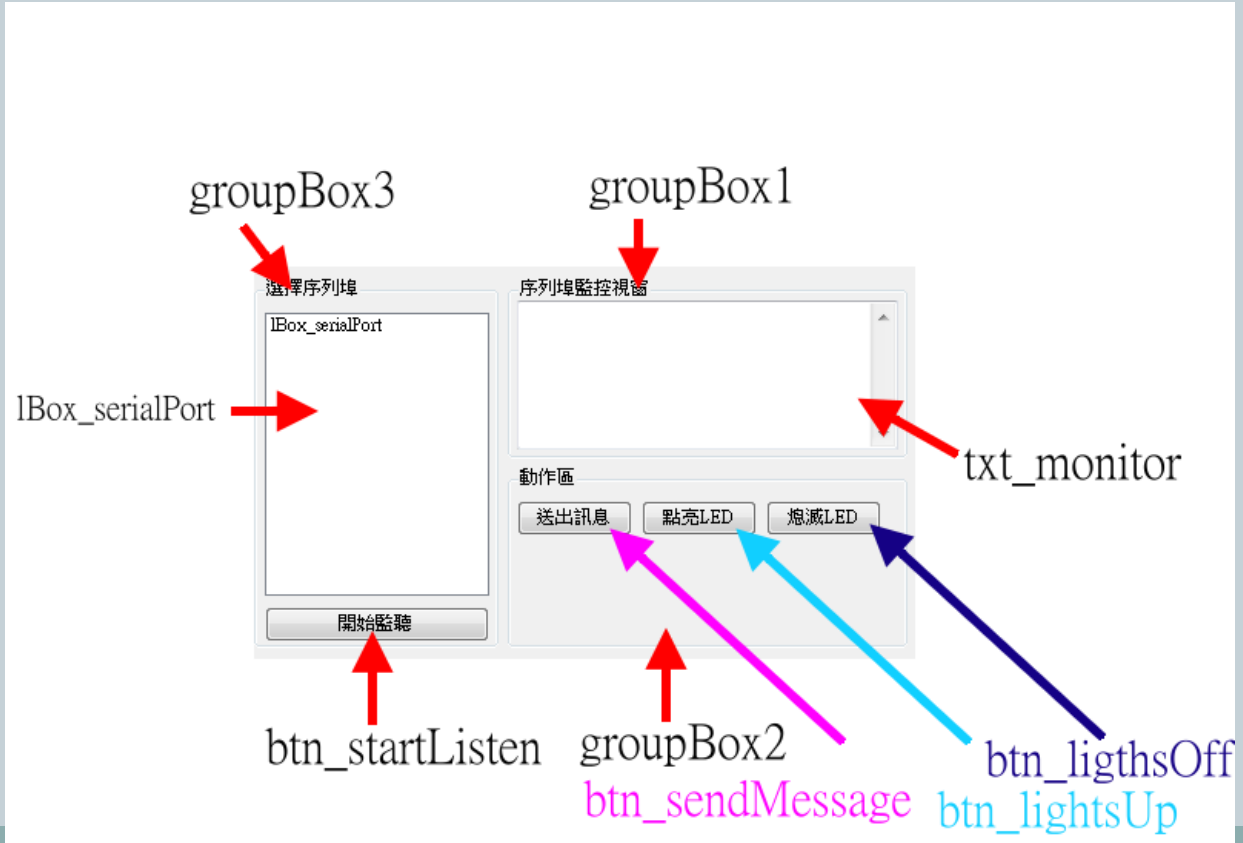
- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 實驗目的：
    - 在Arduino UNO上建立一個會持續發出序列埠訊息的序列埠程式，除此之外我們再增加一段功能讓Arduino UNO具有接收序列埠訊息的能力，並且能夠依照接收到的指令來點亮LED或是熄滅LED。
    - 我們暫時先不討論在Arduino UNO上安裝感測器元件。
    - 接下來我們會在電腦端設計一個應用程式來接收Arduino UNO所發出的序列埠訊息，除此之外我們再增加一段功能讓電腦端的應用程式具有傳送序列埠訊息的能力，並且能夠傳送指令來點亮Arduino UNO的LED或是熄滅LED。
  - 所需開發軟體：
    - Microsoft Visual Studio 2010 C#
    - Arduino IDE

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - Microsoft Visual Studio 2010 C#部分：
    - 執行Microsoft Visual Studio 2010我們按下「新增專案」的按鈕（或者是按下鍵盤的「Control + Shift + N」）來新增一個「Visual C#」的「Windows Form應用程式」。
    - 應用程式的框架版本可以選擇「.NET Framework 4」即可。最後就是指定這個專案的「名稱」、「位置」以及「方案名稱」，請依照自己的需求來命名。
    - 在我們選擇好「.NET Framework 4」並且指定專案的「名稱」、「位置」以及「方案名稱」之後，我們就可以按下「確定」就可以建立一個「Windows Form應用程式」了。
    - 建立好專案後，我們需要先建立「Windows Form應用程式」的介面。

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」的介面，介面編排如下列圖示：



4-5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」的介面，介面編排說明如下列所示：
    - Form１ 大小：４５０px * ３００px。
    - Form１ 標題：SERIALPORT MONITOR 。
    - Form１ 事件：Form1_FormClosing、Form1_Load 。
    - Form１ 取消最大化。
    - btn_startListen 大小：１４６px * ２３px。
    - btn_startListen 標題：開始監聽。
    - btn_startListen 事件： btn_startListen_Click。
    - btn_startListen 座標：７px * ２１３px 。
    - lBox_serialPort 大小：１４６px * １８４px。
    - lBox_serialPort 座標：７px * ２２px 。

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」的介面，介面編排說明如下列所示：
    - groupBox１ 大小：２６０px * １１７px。
    - groupBox１ 標題：序列埠監控視窗。
    - groupBox１ 座標：１６６px * １２px 。
    - groupBox２ 大小：２６０px * １１７px。
    - groupBox２ 標題：動作區。
    - groupBox２ 座標：１６６px * １３５px 。
    - groupBox３ 大小：１５９px * ２４０px。
    - groupBox３ 標題：選擇序列埠。
    - groupBox３ 座標：１px * １２px 。

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」的介面，介面編排說明如下列所示：
    - btn_sendMessage 大小：７５px * ２３px。
    - btn_sendMessage 標題：送出訊息。
    - btn_sendMessage 事件： btn_sendMessage_Click。
    - btn_sendMessage 座標：６px * ２１px 。
    - btn_lightsUp 大小：７５px * ２３px。
    - btn_lightsUp 標題：點亮LED。
    - btn_lightsUp 事件： btn_lightsUp_Click。
    - btn_lightsUp 座標：８７px * ２１px 。
    - btn_ligthsOff 大小：７５px * ２３px。
    - btn_ligthsOff 標題：熄滅LED。
    - btn_ligthsOff 事件： btn_ligthsOff_Click。
    - btn_ligthsOff 座標：１６８px * ２１px 。

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」，「 Form1.cs 」的程式碼。
    - using System;
    - using System.Collections.Generic;
    - using System.ComponentModel;
    - using System.Data;
    - using System.Drawing;
    - using System.Linq;
    - using System.Text;
    - using System.Windows.Forms;
    - using System.IO.Ports; //必須引用System.IO.Ports才可以直接找到SerialPort。

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」，「 Form1.cs 」的程式碼。

```
namespace CS_WFAPP_SERIALPORT_MONITOR
{
    public partial class Form1 : Form
    {
        //定義一個序列埠元件。
        protected SerialPort serialPort = new SerialPort();
       //定義一個計時器元件。
        protected Timer timer = new Timer();
public Form1()
        {
            InitializeComponent();
        }
```

4-5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」，「 Form1.cs 」的程式碼。

```
//Form產生後的進入點事件。
private void Form1_Load(object sender, EventArgs e)
        {
            //將系統目前連接的序列埠列表交給lBox_serialPort作為資料來源列表出來。
            lBox_serialPort.DataSource = SerialPort.GetPortNames();
            //設定計時器元件的觸發間隔時間，１０００毫秒等於１秒。
            //設定觸發間隔時間為１.５秒或是２秒，能夠改善程式延滯的情形。
            timer.Interval = 1500;
            //設定計時器元件的觸發事件。
            timer.Tick += new EventHandler(timer_Tick);
            //取消送出訊息的致能。
            btn_sendMessage.Enabled = false;
            //取消點亮LED的致能。
            btn_lightsUp.Enabled = false;
            //取消熄滅LED的致能。
            btn_ligthsOff.Enabled = false;
        }
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」，「 Form1.cs 」的程式碼。

```
//計時器元件的觸發事件
private void timer_Tick(object sender, EventArgs e) {
        try
        {
            //將序列埠元件所讀取到的資料行再加上一個換行符號，然後交給txt_monitor顯示。
            txt_monitor.Text += serialPort.ReadLine() + System.Environment.NewLine;
        }
        catch { }
    }
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」，「 Form1.cs 」的程式碼。

```
//Form關閉時的事件
        private void Form1_FormClosing(object sender, FormClosingEventArgs e)
        {
            //判斷序列埠連線狀態是否為連線中
            if (serialPort.IsOpen)
            {
                //關閉連線
                serialPort.Close();
            }
            //釋放序列埠元件
            serialPort.Dispose();
        }
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」，「 Form1.cs 」的程式碼。

```
//按下開始監聽/停止監聽按鈕的事件。
private void btn_startListen_Click(object sender, EventArgs e)
  {
    //當按鈕的標題為停止監聽時。
    if (btn_startListen.Text == "停止監聽")
    {          //當序列埠列表致能被取消時。
      if (lBox_serialPort.Enabled == false)
      {        //將序列埠列表設為致能。
        lBox_serialPort.Enabled = true;
        //將groupBox1的標題回復成「序列埠監控視窗」。
        groupBox1.Text = "序列埠監控視窗";
        try
        {   //判斷序列埠連線狀態是否為連線中。
          if (serialPort.IsOpen)
          {    //關閉連線。
            serialPort.Close();
          }
        }
        catch
        { }
        //停止計時器元件。
        timer.Enabled = false;
        //將btn_startListen標題回復為「開始監聽」。
        btn_startListen.Text = "開始監聽";
        //將送出訊息設為致能。
        btn_sendMessage.Enabled = false;
        //將點亮LED設為致能。
        btn_lightsUp.Enabled = false;
        //將熄滅LED設為致能。
        btn_ligthsOff.Enabled = false;
      }
    }
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」，「 Form1.cs 」的程式碼。

```
//當按鈕狀態為開始監聽時。
        else
        {           //當序列埠列表有選擇一個項目時。
          if (lBox_serialPort.SelectedIndex > -1)
          {         //將序列埠列表所選的項目作為序列埠元件監聽的對象。
            serialPort.PortName = lBox_serialPort.SelectedValue.ToString();
            //設定序列埠元件的傳輸鮑率為９６００，要跟Arduino的傳輸鮑率一樣。
            serialPort.BaudRate = 9600;
            //將groupBox1的標題設定為「序列埠監控視窗 - 連接埠名稱」提示使用者目前所選的連接埠是什麼。
            groupBox1.Text = string.Format("序列埠監控視窗 - {0}", lBox_serialPort.SelectedValue.ToString());
            //將txt_monitor的內容設定為「序列埠監控視窗 - 連接埠名稱」再加上一個換行符號，提示使用者目前所選的連接埠是什麼。
            txt_monitor.Text = string.Format("選擇序列埠：{0}，開始監控。{1}", lBox_serialPort.SelectedValue.ToString(), System.Environment.NewLine);
            //取消序列埠列表的致能。
            lBox_serialPort.Enabled = false;
            try  {                  //判斷序列埠連線狀態是否為關閉中。
              if (!serialPort.IsOpen)
              {                 //開啟連線。
                serialPort.Open();
              }
            }
            catch    { }
            //開啟計時器元件。
            timer.Enabled = true;
            //將btn_startListen標題設定為「停止監聽」。
            btn_startListen.Text = "停止監聽";
            //取消送出訊息的致能。
            btn_sendMessage.Enabled = true;
            //取消點亮LED的致能。
            btn_lightsUp.Enabled = true;
            //取消熄滅LED的致能。
            btn_ligthsOff.Enabled = true;
          }
        }
      }
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」，「 Form1.cs 」的程式碼。

```
//送出訊息事件。
    private void btn_sendMessage_Click(object sender, EventArgs e)
    {
      try
      {
        //判斷序列埠連線狀態是否為關閉中。
        if (!serialPort.IsOpen)
        {
          //開啟連線。
          serialPort.Open();
        }

        //由電腦端傳送一個序列埠訊息給Arduino。
        serialPort.WriteLine("Good Day");
      }
      catch
      { }
    }
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」，「 Form1.cs 」的程式碼。

```
//點亮ＬＥＤ事件。
    private void btn_lightsUp_Click(object sender, EventArgs e)
    {
      try
      {
        //判斷序列埠連線狀態是否為關閉中。
        if (!serialPort.IsOpen)
        {
          //開啟連線。
          serialPort.Open();
        }

        //由電腦端傳送一個序列埠訊息給Arduino，命令Arduino點亮ＬＥＤ。
        serialPort.Write ("$ON");
      }
      catch
      { }
    }
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」的程式碼，「 Form1.cs 」的程式碼。

```
//熄滅ＬＥＤ事件。
        private void btn_ligthsOff_Click(object sender, EventArgs e)
        {
            try
            {
                //判斷序列埠連線狀態是否為關閉中。
                if (!serialPort.IsOpen)
                {
                    //開啟連線。
                    serialPort.Open();
                }

                //由電腦端傳送一個序列埠訊息給Arduino，命令Arduino熄滅ＬＥＤ。
                serialPort.Write("$OFF");
            }
            catch
            { }
        }

    }
}
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」，「Form1.Designer.cs」的程式碼。

```
namespace CS_WFAPP_SERIALPORT_MONITOR
{
  partial class Form1
  {
    /// <summary>
    /// 設計工具所需的變數。
    /// </summary>
    private System.ComponentModel.IContainer components = null;

    /// <summary>
    /// 清除任何使用中的資源。
    /// </summary>
    /// <param name="disposing">如果應該處置 Managed 資源則為 true，否則為 false。</param>
    protected override void Dispose(bool disposing)
    {
      if (disposing && (components != null))
      {
        components.Dispose();
      }
      base.Dispose(disposing);
    }

    #region Windows Form 設計工具產生的程式碼
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」，「Form1.Designer.cs」的程式碼。

```
/// <summary>
        /// 此為設計工具支援所需的方法 - 請勿使用程式碼編輯器
        /// 修改這個方法的內容。
        /// </summary>
        private void InitializeComponent()
        {
            this.txt_monitor = new System.Windows.Forms.TextBox();
            this.groupBox1 = new System.Windows.Forms.GroupBox();
            this.groupBox2 = new System.Windows.Forms.GroupBox();
            this.btn_ligthsOff = new System.Windows.Forms.Button();
            this.btn_lightsUp = new System.Windows.Forms.Button();
            this.btn_sendMessage = new System.Windows.Forms.Button();
            this.groupBox3 = new System.Windows.Forms.GroupBox();
            this.btn_startListen = new System.Windows.Forms.Button();
            this.lBox_serialPort = new System.Windows.Forms.ListBox();
            this.groupBox1.SuspendLayout();
            this.groupBox2.SuspendLayout();
            this.groupBox3.SuspendLayout();
            this.SuspendLayout();
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」，「Form1.Designer.cs」的程式碼。

```
//
// txt_monitor
//
this.txt_monitor.Location = new System.Drawing.Point(6, 14);
this.txt_monitor.Multiline = true;
this.txt_monitor.Name = "txt_monitor";
this.txt_monitor.ScrollBars = System.Windows.Forms.ScrollBars.Both;
this.txt_monitor.Size = new System.Drawing.Size(248, 97);
this.txt_monitor.TabIndex = 0;
//
// groupBox1
//
this.groupBox1.Controls.Add(this.txt_monitor);
this.groupBox1.Location = new System.Drawing.Point(166, 12);
this.groupBox1.Name = "groupBox1";
this.groupBox1.Size = new System.Drawing.Size(260, 117);
this.groupBox1.TabIndex = 1;
this.groupBox1.TabStop = false;
this.groupBox1.Text = "序列埠監控視窗";
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」，「Form1.Designer.cs」的程式碼。

```
            //
            // groupBox2
            //
            this.groupBox2.Controls.Add(this.btn_ligthsOff);
            this.groupBox2.Controls.Add(this.btn_lightsUp);
            this.groupBox2.Controls.Add(this.btn_sendMessage);
            this.groupBox2.Location = new System.Drawing.Point(166, 135);
            this.groupBox2.Name = "groupBox2";
            this.groupBox2.Size = new System.Drawing.Size(260, 117);
            this.groupBox2.TabIndex = 2;
            this.groupBox2.TabStop = false;
            this.groupBox2.Text = "動作區";
            //
            // btn_ligthsOff
            //
            this.btn_ligthsOff.Location = new System.Drawing.Point(168, 21);
            this.btn_ligthsOff.Name = "btn_ligthsOff";
            this.btn_ligthsOff.Size = new System.Drawing.Size(75, 23);
            this.btn_ligthsOff.TabIndex = 2;
            this.btn_ligthsOff.Text = "熄滅LED";
            this.btn_ligthsOff.UseVisualStyleBackColor = true;
            this.btn_ligthsOff.Click += new System.EventHandler(this.btn_ligthsOff_Click);
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」，「Form1.Designer.cs」的程式碼。

```
//
// btn_lightsUp
//
this.btn_lightsUp.Location = new System.Drawing.Point(87, 21);
this.btn_lightsUp.Name = "btn_lightsUp";
this.btn_lightsUp.Size = new System.Drawing.Size(75, 23);
this.btn_lightsUp.TabIndex = 1;
this.btn_lightsUp.Text = "點亮LED";
this.btn_lightsUp.UseVisualStyleBackColor = true;
this.btn_lightsUp.Click += new System.EventHandler(this.btn_lightsUp_Click);
//
// btn_sendMessage
//
this.btn_sendMessage.Location = new System.Drawing.Point(6, 21);
this.btn_sendMessage.Name = "btn_sendMessage";
this.btn_sendMessage.Size = new System.Drawing.Size(75, 23);
this.btn_sendMessage.TabIndex = 0;
this.btn_sendMessage.Text = "送出訊息";
this.btn_sendMessage.UseVisualStyleBackColor = true;
this.btn_sendMessage.Click += new System.EventHandler(this.btn_sendMessage_Click);
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」，「Form1.Designer.cs」的程式碼。

```
            //
            // groupBox3
            //
            this.groupBox3.Controls.Add(this.btn_startListen);
            this.groupBox3.Controls.Add(this.lBox_serialPort);
            this.groupBox3.Location = new System.Drawing.Point(1, 12);
            this.groupBox3.Name = "groupBox3";
            this.groupBox3.Size = new System.Drawing.Size(159, 240);
            this.groupBox3.TabIndex = 3;
            this.groupBox3.TabStop = false;
            this.groupBox3.Text = "選擇序列埠";
            //
            // btn_startListen
            //
            this.btn_startListen.Location = new System.Drawing.Point(7, 213);
            this.btn_startListen.Name = "btn_startListen";
            this.btn_startListen.Size = new System.Drawing.Size(146, 23);
            this.btn_startListen.TabIndex = 1;
            this.btn_startListen.Text = "開始監聽";
            this.btn_startListen.UseVisualStyleBackColor = true;
            this.btn_startListen.Click += new System.EventHandler(this.btn_startListen_Click);
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」，「Form1.Designer.cs」的程式碼。

```
            //
            // lBox_serialPort
            //
            this.lBox_serialPort.FormattingEnabled = true;
            this.lBox_serialPort.ItemHeight = 12;
            this.lBox_serialPort.Location = new System.Drawing.Point(7,
22);
            this.lBox_serialPort.Name = "lBox_serialPort";
            this.lBox_serialPort.Size = new System.Drawing.Size(146, 184);
            this.lBox_serialPort.TabIndex = 0;
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」，「Form1.Designer.cs」的程式碼。

```
            //
            // Form1
            //
            this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 12F);
            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
            this.ClientSize = new System.Drawing.Size(434, 262);
            this.Controls.Add(this.groupBox3);
            this.Controls.Add(this.groupBox2);
            this.Controls.Add(this.groupBox1);
            this.MaximizeBox = false;
            this.Name = "Form1";
            this.Text = "SERIALPORT MONITOR";
            this.FormClosing += new System.Windows.Forms.FormClosingEventHandler(this.Form1_FormClosing);
            this.Load += new System.EventHandler(this.Form1_Load);
            this.groupBox1.ResumeLayout(false);
            this.groupBox1.PerformLayout();
            this.groupBox2.ResumeLayout(false);
            this.groupBox3.ResumeLayout(false);
            this.ResumeLayout(false);

        }

        #endregion
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Windows Form應用程式」，「Form1.Designer.cs」的程式碼。

```
private System.Windows.Forms.TextBox txt_monitor;
        private System.Windows.Forms.GroupBox groupBox1;
        private System.Windows.Forms.GroupBox groupBox2;
        private System.Windows.Forms.GroupBox groupBox3;
        private System.Windows.Forms.ListBox lBox_serialPort;
        private System.Windows.Forms.Button btn_startListen;
        private System.Windows.Forms.Button btn_ligthsOff;
        private System.Windows.Forms.Button btn_lightsUp;
        private System.Windows.Forms.Button btn_sendMessage;
    }
}
```

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Arduino」的程式碼：

```
//定義13腳為LEDPIN。
#define LEDPIN 13

//定義LED的狀態為低電位。
int LEDPINSTATE = 0;

//Arduino程式的初始設定, 僅於上電時執行一次。
void setup(){
 //設定傳輸率為9600。
 Serial.begin(9600);
 //指定LEDPIN(第13腳位)為出。
 pinMode(LEDPIN , OUTPUT);
}

//Arduino程式的迴圈區段, 一直到斷電才結束動作。
void loop() {
 //建立一個收集字元的字串。
 String command;
 //建立一個while迴圈, 收集序列埠的訊息。
 while( Serial.available() )
 {
  //延遲一微秒, 防止接收資料出錯。
  delay(1);
  //當序列埠有資料的時候就開始將資料收進來。
  if (Serial.available() > 0)
  {
   //設定一個字元變數來逐一接收序列埠的訊息。
   char c = Serial.read();
   //將逐一收到的序列埠訊息存入字串中。
   command += c;
  }
 }
```

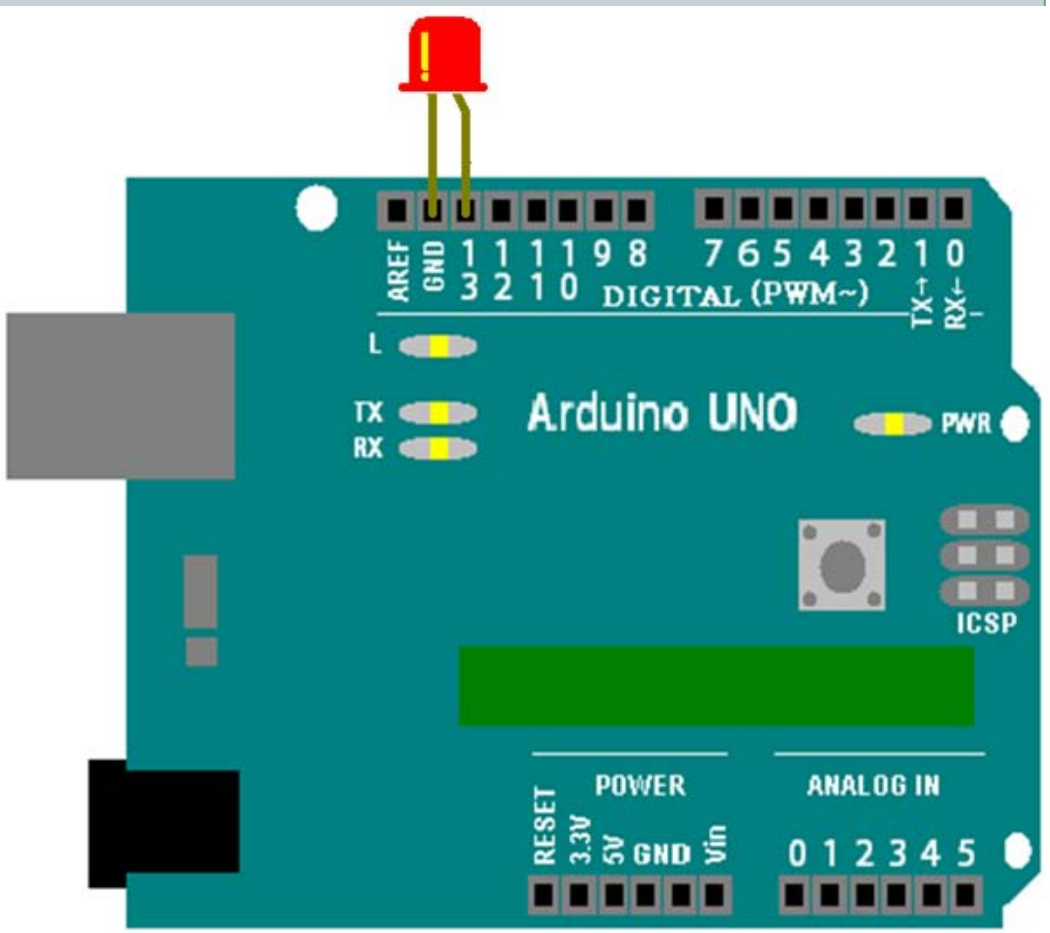
4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Arduino」的程式碼：

```
//當字串有內容時開始判斷字串是否有包含點亮ＬＥＤ或是熄滅ＬＥＤ的指令。
if (command.length() > 0)
{
 //當字串中是為點亮ＬＥＤ的指令。
 if (command == "$ON")
 {
  //將LED的狀態設定為高電位。
  LEDPINSTATE = 1;
 }
 //當字串中是為熄滅ＬＥＤ的指令。
 else if (command == "$OFF")
 {
  //將LED的狀態設定為低電位。
  LEDPINSTATE = 0;
 }
 //不是指令的時候就將訊息列印出來。
 else
 {
  //列出從電腦端接收到的訊息。
  Serial.println("Command from PC:" + command);
 }
}
//當字串為0, 就持續送出序列埠訊息。
else
{
 //印出Hello World!, 可以在序列埠監控視窗看到。
 Serial.println("Hello World!");
}
 //根據LEDPINSTATE的狀態來決定LED燈的亮滅。
digitalWrite(LEDPIN, LEDPINSTATE);
// 延遲１秒再進行一次動作。
delay(1000);
}
```

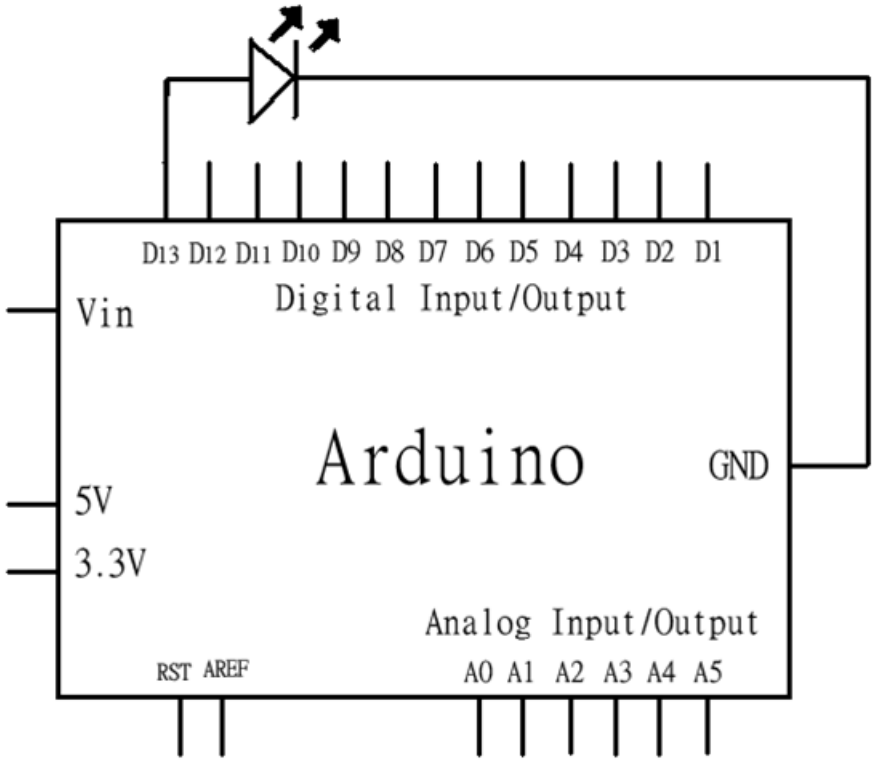
4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Arduino」的接線配置方式：
    - 將LED較短的針腳(陰極)接於DIGITAL (PWM~)的Pin GND，另外一隻較長的針腳(陽極)接於DIGITAL (PWM~) 的Pin 13 即可。
    - 如右圖所示：



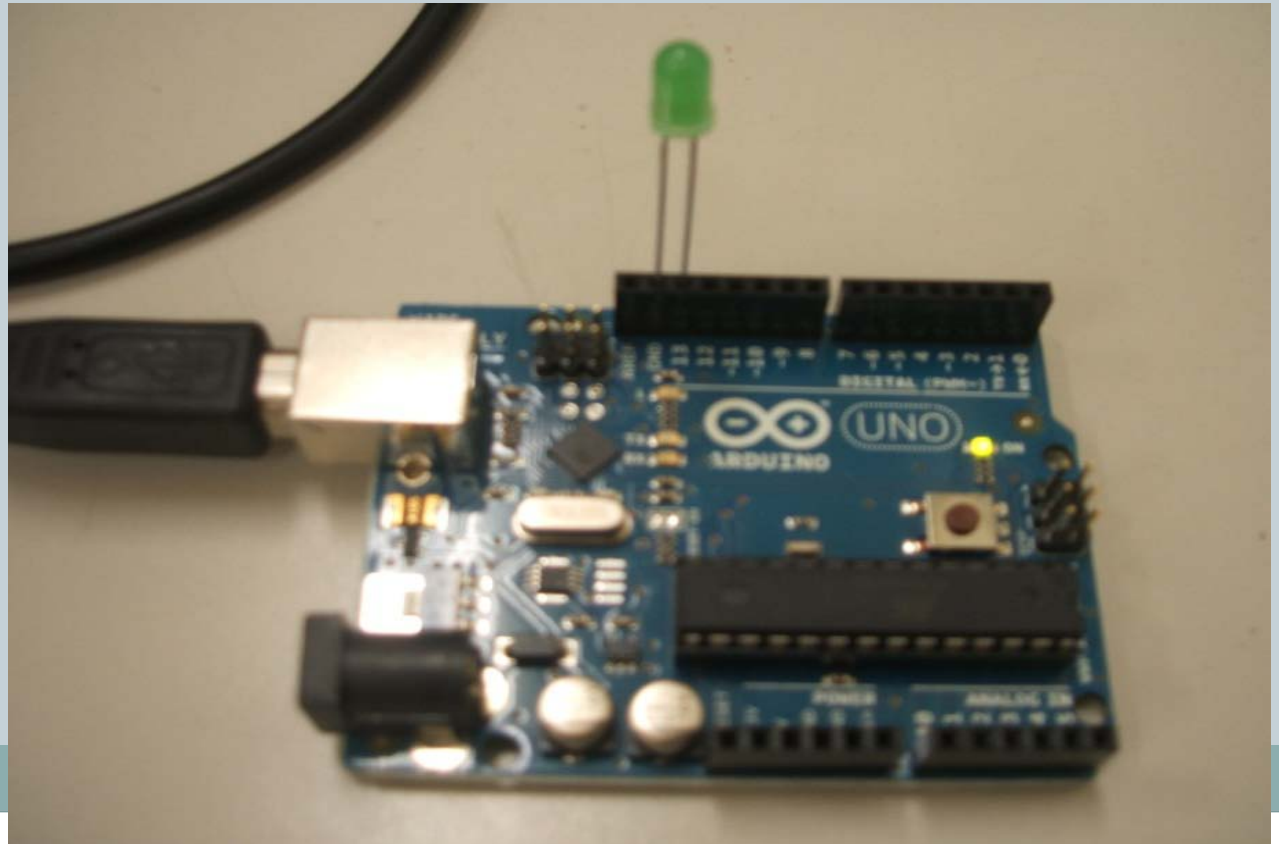
4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Arduino」的電路圖：
    - 將LED較短的針腳(陰極)接於DIGITAL (PWM~)的Pin GND，另外一隻較長的針腳(陽極)接於DIGITAL (PWM~) 的Pin 13 即可。
    - 如右圖所示：



4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 建立「Arduino」的接線圖：LED的長腳接到１３腳位，短腳接到隔壁的GND腳位。



4 - 5 Arduino與電腦端互動設計

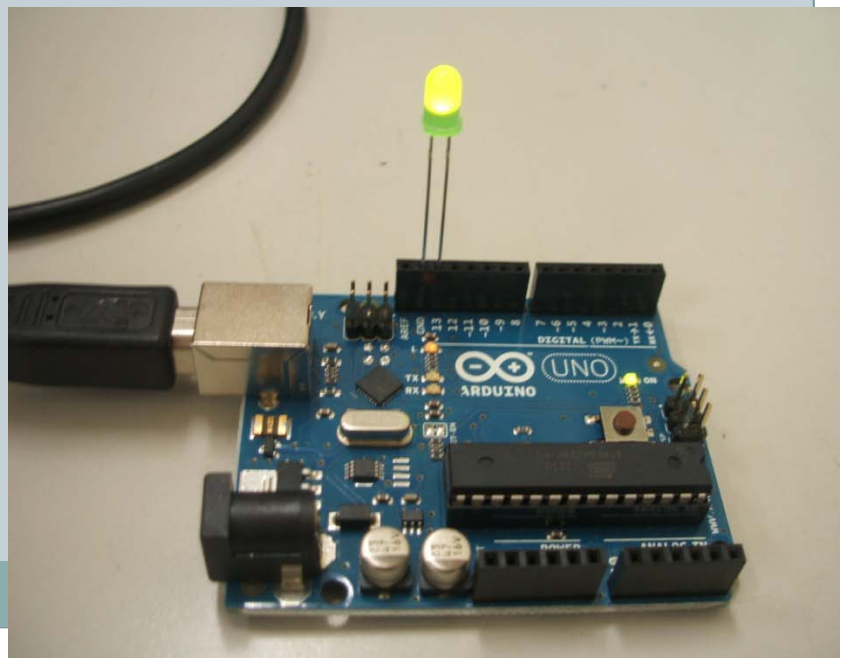
- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 動作說明：
    - 「Arduino」撰寫完成後請將程式「上傳」至Arduino UNO。
    - 「Windows Form應用程式」介面以及程式碼撰寫完成之後，請將程式「編譯」，沒有錯誤之後就可以「開始偵錯」。
    - 在「Windows Form應用程式」介面中選擇Arduino UNO所在的序列埠，然後按下「開始監聽」，可以看到序列埠監控視窗開始接收到訊息。
    - 訊息如下列所示：
      - 選擇序列埠：COM4，開始監控。
      - Hello World!
      - Hello World!
      - Hello World!
      - Hello World!
      - Hello World!
      - Hello World!
      - Hello World!
      - ...

4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 動作說明：
    - 在「Windows Form應用程式」介面中按下「送出訊息」，可以看到序列埠監控視窗開始接收到訊息。
    - 訊息如下列所示：
      - 選擇序列埠：COM4，開始監控。
      - Hello World!
      - Command from PC:Good Day
      - Hello World!
      - Command from PC:Good Day
      - Hello World!
      - Command from PC:Good Day
      - Hello World!
      - Command from PC:Good Day
      - ...

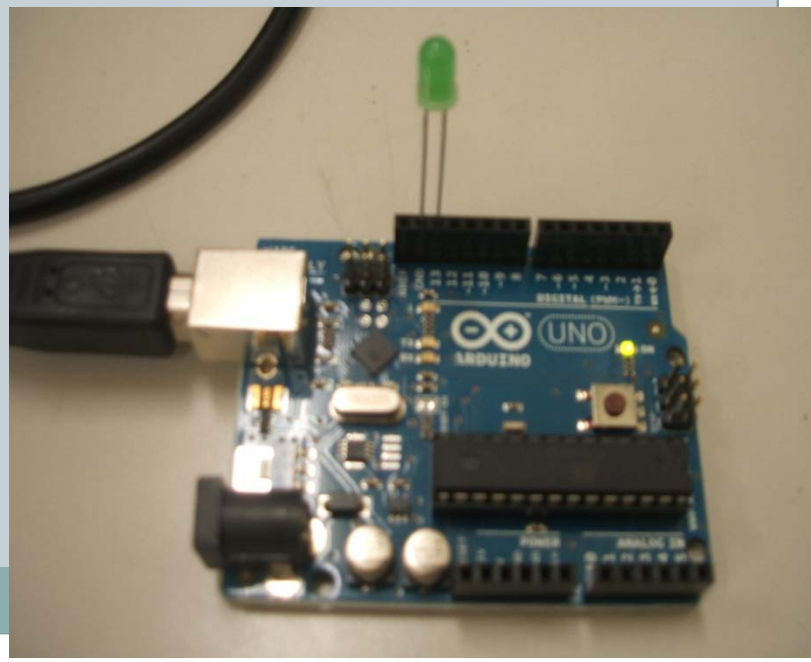
4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 動作說明：
    - 在「Windows Form應用程式」介面中按下「點亮LED」，可以看到Arduino UNO LED燈被點亮，同時可以發現在Arduino UNO 上的L LED燈也點亮了，這是因為Arduino UNO L LED燈與第１３腳位的狀態共用。



4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 動作說明：
    - 在「Windows Form應用程式」介面中按下「熄滅LED」，可以看到Arduino UNO LED燈熄滅，同時可以發現在Arduino UNO 上的LLED燈也熄滅了，這是因為Arduino UNO L LED燈與第１３腳位的狀態共用。



4 - 5 Arduino與電腦端互動設計

- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 動作說明：
    - 觀察「Windows Form應用程式」介面中的「序列埠監控視窗」是否一秒接收到一次Hello World!的訊息，並且持續不中斷。
    - 觀察「Windows Form應用程式」介面中按下「開始監聽」後，其按鈕的標題是否轉為「停止監聽」，並且無法選擇序列埠。
    - 觀察「Windows Form應用程式」介面中按下「停止監聽」後，其按鈕的標題是否轉為「開始監聽」，並且可以選擇序列埠，而「序列埠監控視窗」是否停止監聽來自Arduino的訊息。

4-5 Arduino與電腦端互動設計

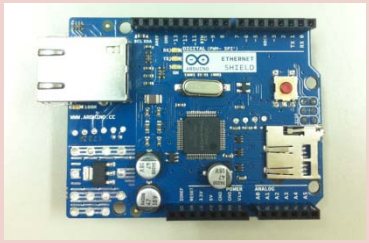
- 實驗名稱：HELLO WORLD WITH COMMAND。
  - 實驗結果：
    - 經過這個實驗後，我們就能夠自行撰寫監聽序列埠與送出序列埠的程式。
    - 我們可以監聽Arduino並且依照監聽到的訊息來做情境模擬。
    - 例如：讓Arduino裝上超音波感測器元件，當超音波感測器元件感測到有物體接近時可以發送訊一個序列埠訊息讓電腦端接收到，而電腦端就可以根據這個訊息來分析狀況。
    - 我們可以送出序列埠訊息給Arduino並且依照送出的序列埠訊息來做情境模擬。
    - 例如：讓Arduino裝上超音波感測器元件，並且由電腦端控制超音波感測器元件的開或關，當電腦端指定超音波感測器元件為開啟的時候，超音波感測器元件感測到有物體接近時可以發送訊一個序列埠訊息讓電腦端接收到，而電腦端就可以根據這個訊息來分析狀況。

4-6 Arduino與行動裝置互動設計

- 這節介紹的是Arduino與行動裝置互動設計，我們將學習如何撰寫行動裝置的程式來與Arduino UNO搭配感測器元件(Sensor)的互動實驗。
- 本節的目的如下：
  - 使用Microsoft Visual Studio 2010搭配Windows Phone SDK 7.1撰寫行動裝置程式。
  - 動手撰寫Arduino程式驅動感測器元件。
  - 動手接線瞭解簡單的電子電路。
  - 以電腦端程式來與Arduino進行互動。
  - 藉由簡單的實作得到更多的發想，期望能夠從實驗中組合出實用有趣的應用

4-6 Arduino與行動裝置互動設計

- 實驗名稱：Arduino 控制LED亮滅實驗。
  - 所需硬體材料：

元件實體圖	元件數量	元件名稱
	1	ＬＥＤ燈（任何顏色皆可）
	1	Arduino UNO控制板
	1	Arduino Ethernet Shield R３乙太網卡擴充板

4-6 Arduino與行動裝置互動設計

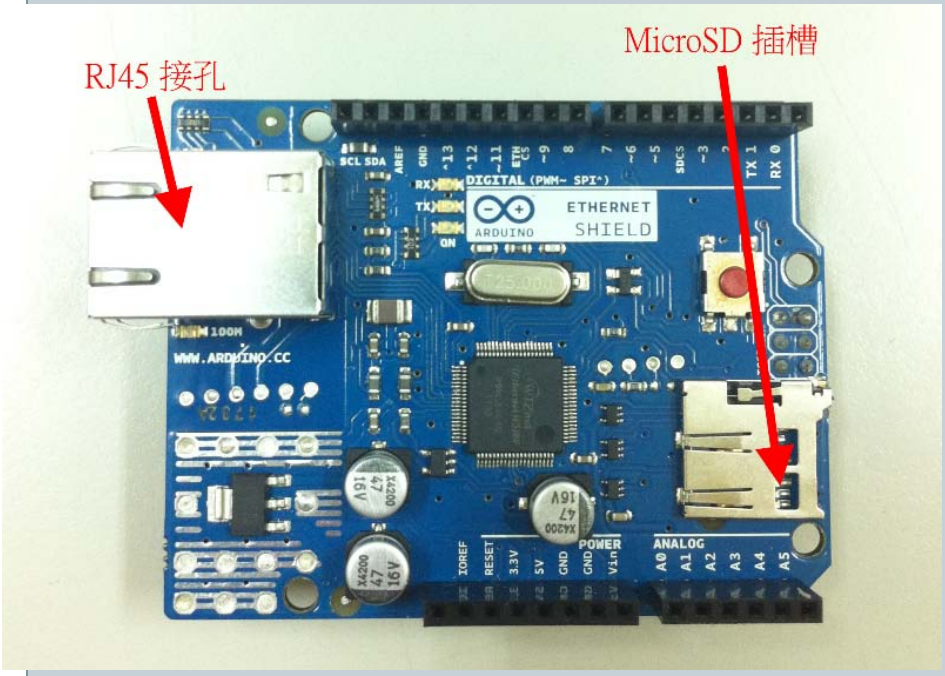
- 實驗名稱：Arduino 控制LED亮滅實驗。
  - 實驗目的：
    - 在Arduino UNO上建立一個負責接收連入訊息的網頁伺服器，這個網頁伺服器會接收HTTP GET的網址中是否帶有指令？1或者是？0，然後再依照指令來執行點亮LED或是熄滅LED的動作。
    - 我們暫時先不討論在Arduino UNO上安裝感測器元件。
    - 接下來我們會在電腦端設計一個應用程式來接收Arduino UNO所發出的序列埠訊息，除此之外我們再增加一段功能讓電腦端的應用程式具有傳送序列埠訊息的能力，並且能夠傳送指令來點亮Arduino UNO的LED或是熄滅LED。
  - 所需開發軟體：
    - Microsoft Visual Studio 2010 C#
    - Windows Phone SDK 7.1
    - Arduino IDE
    - 若要使用Windows Phone作為測試專案的目標，則需要安裝Microsoft Zune。

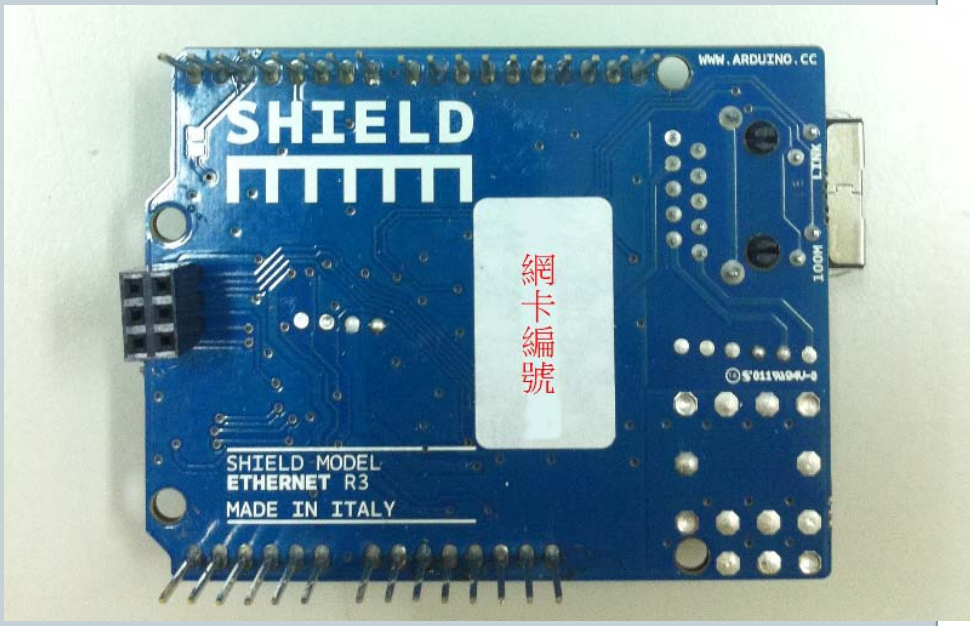
4-4 Arduino裝置人機互動設計

- 實驗名稱：Arduino 控制LED亮滅實驗。
  - Arduino 擴展板名稱：Arduino 乙太網路擴展板。
  - Arduino 擴展板型號：Arduino Ethernet Shield R３。
  - Arduino 擴展板規格描述：
    - 這是一款能讓Arduino UNO具有連接乙太網路（10/100Mb）能力的擴展板，安裝方法很簡單，只要堆疊上Arduino UNO即可。
    - 電力來源為Arduino UNO
    - 除了具有一個RJ４５網路孔讓使用者連接乙太網路線之外， Arduino乙太網路擴展板上還有一個MicroSD卡插槽可以用來讀寫記憶卡中的資料。
    - 在撰寫Arduino程式的時候必須引用下列兩個.h檔。
      - #include <SPI.h>
      - #include <Ethernet.h>

4-4 Arduino裝置人機互動設計

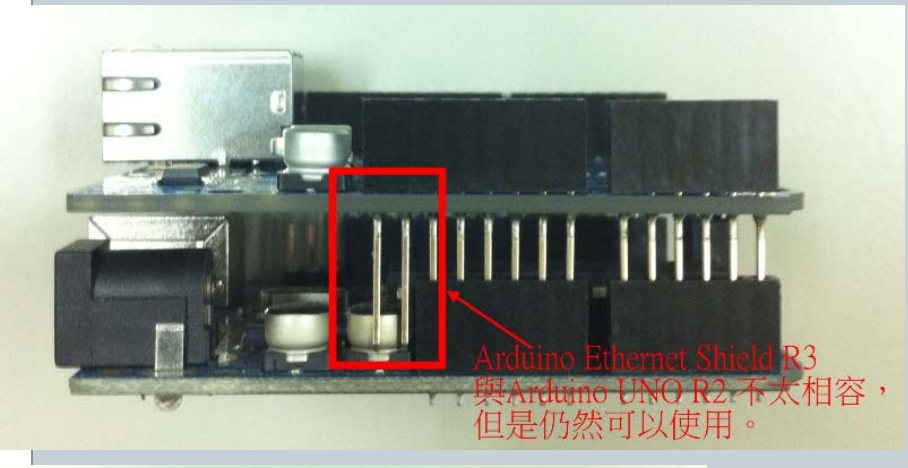
- 實驗名稱：Arduino 控制LED亮滅實驗。
  - 下列為Arduino 乙太網路擴展板之實體圖：

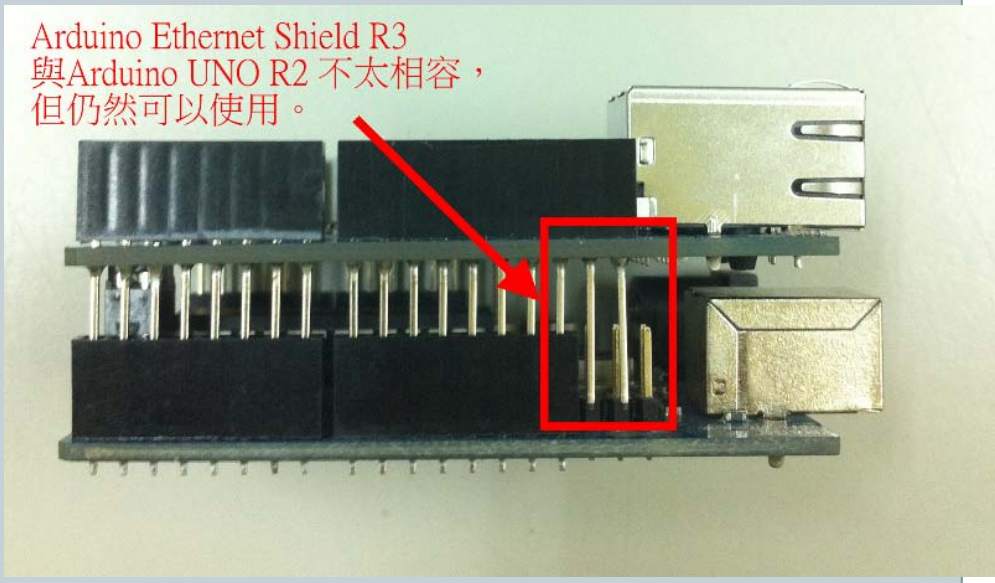


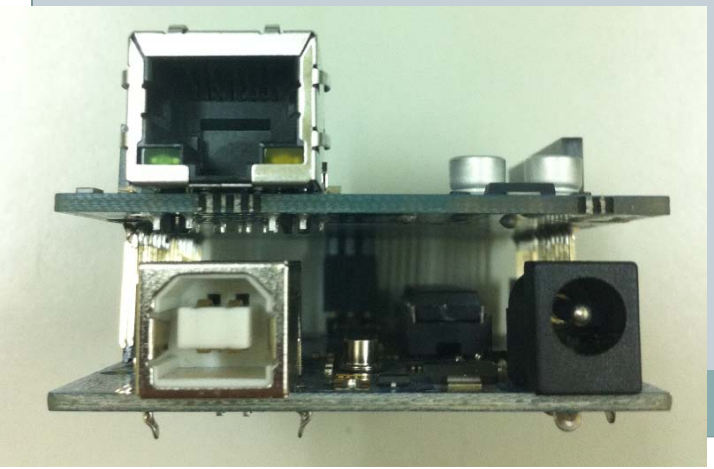


4-4 Arduino裝置人機互動設計

- 實驗名稱：Arduino 控制LED亮滅實驗。
  - 安裝方向請依照下圖所示。

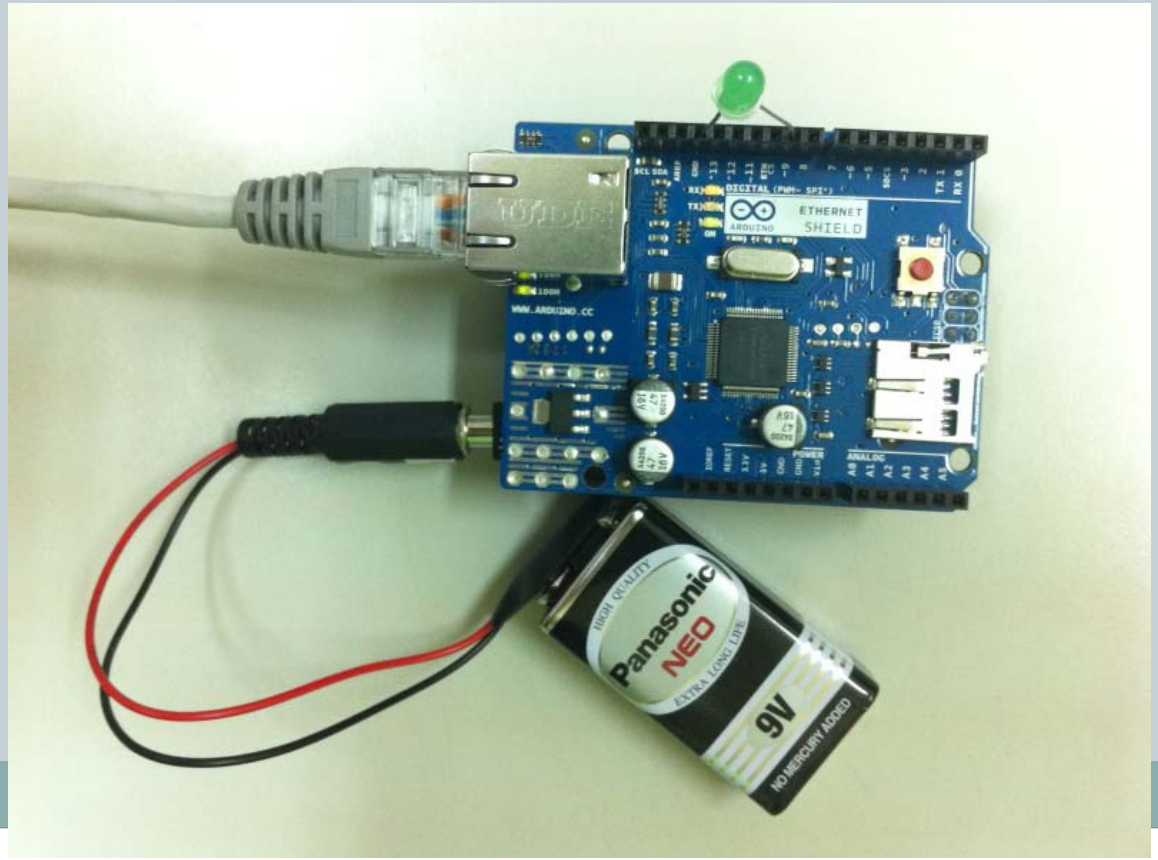






4-4 Arduino裝置人機互動設計

- 實驗名稱：Arduino 控制LED亮滅實驗。
  - 也可以使用９V電池驅動Arduino UNO + Arduino Ethernet Shield。



4-6 Arduino與行動裝置互動設計

- 實驗名稱：Arduino 控制LED亮滅實驗。
  - Microsoft Visual Studio 2010 C#部分：
    - 執行Microsoft Visual Studio 2010我們按下「新增專案」的按鈕（或者是按下鍵盤的「Control + Shift + N」）來新增一個「Visual C#」的「Windows Phone 應用程式」。
    - 應用程式的框架版本可以選擇「.NET Framework 4」即可。最後就是指定這個專案的「名稱」、「位置」以及「方案名稱」，請依照自己的需求來命名。
    - 在我們選擇好「.NET Framework 4」並且指定專案的「名稱」、「位置」以及「方案名稱」之後，我們就可以按下「確定」就可以建立一個「 Windows Phone 應用程式」了。
    - 建立好專案後，我們需要先建立「 Windows Phone 應用程式」的介面。

4-6 Arduino與行動裝置互動設計

- 實驗名稱：Arduino 控制LED亮滅實驗。
  - 建立「 Windows Phone 應用程式」的介面，介面編排如下列圖示：

4-6 Arduino與行動裝置互動設計

- 實驗名稱：Arduino 控制LED亮滅實驗。
  - 建立「 Windows Phone 應用程式」，「MainPage.xaml」的程式碼。

```
<phone:PhoneApplicationPage
    x:Class="CS_PhoneApp1.MainPage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:phone="clr-namespace:Microsoft.Phone.Controls;assembly=Microsoft.Phone"
    xmlns:shell="clr-namespace:Microsoft.Phone.Shell;assembly=Microsoft.Phone"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    mc:Ignorable="d" d:DesignWidth="480" d:DesignHeight="768"
    FontFamily="{StaticResource PhoneFontFamilyNormal}"
    FontSize="{StaticResource PhoneFontSizeNormal}"
    Foreground="{StaticResource PhoneForegroundBrush}"
    SupportedOrientations="Portrait" Orientation="Portrait"
    shell:SystemTray.IsVisible="True">
```

4-6 Arduino與行動裝置互動設計

- 實驗名稱：Arduino 控制LED亮滅實驗。
  - 建立「 Windows Phone 應用程式」，「MainPage.xaml」的程式碼。

```
<!--LayoutRoot 是放置所有頁面的根資料格-->
<Grid x:Name="LayoutRoot" Background="Transparent">
    <Grid.RowDefinitions>
        <RowDefinition Height="Auto"/>
        <RowDefinition Height="*"/>
    </Grid.RowDefinitions>

    <!--TitlePanel 包含應用程式的名稱和頁面標題-->
    <StackPanel x:Name="TitlePanel" Grid.Row="0" Margin="12,17,0,28">
        <TextBlock x:Name="ApplicationTitle" Text="Arduino 控制LED亮滅實驗" Style="{StaticResource PhoneTextNormalStyle}"/>
    </StackPanel>
```

4-6 Arduino與行動裝置互動設計

- 實驗名稱：Arduino 控制LED亮滅實驗。
  - 建立「 Windows Phone 應用程式」，「MainPage.xaml」的程式碼。

```
        <!--ContentPanel - 其他內容置於此-->
        <Grid x:Name="ContentPanel" Grid.Row="1" Margin="12,0,12,0">
            <Button Content="點亮LED" Height="150" HorizontalAlignment="Left" Margin="0,259,0,0" Name="btn_lightsUp" VerticalAlignment="Top" Width="460" Foreground="MidnightBlue" Background="Yellow" Click="btn_lightsUp_Click"></Button>
            <Button Content="熄滅LED" Height="150" HorizontalAlignment="Left" Margin="0,0,0,146" Name="btn_lightsOff" VerticalAlignment="Bottom" Width="460" Background="MidnightBlue" Foreground="Yellow" Click="btn_lightsOff_Click" />
            <TextBlock Text="分配給Arduino Ethernet Shield網卡的IP位址：" Height="30" HorizontalAlignment="Left" Margin="12,31,0,0" Name="textBlock1" VerticalAlignment="Top" />
            <TextBox Height="72" HorizontalAlignment="Left" Margin="-4,67,0,0" Name="txt_ipAddress" Text="127.0.0.1" VerticalAlignment="Top" Width="460" />
            <TextBlock Height="30" HorizontalAlignment="Left" Margin="12,573,0,0" Name="txt_infomation" VerticalAlignment="Top" FontSize="24" />
            <TextBlock Text="Arduino程式碼所指定的網頁伺服器Port：" Height="30" HorizontalAlignment="Left" Margin="12,145,0,0" Name="textBlock2" VerticalAlignment="Top" />
            <TextBox Height="72" HorizontalAlignment="Left" Margin="0,181,0,0" Name="txt_serverPort" Text="80" VerticalAlignment="Top" Width="460" />
        </Grid>
    </Grid>

</phone:PhoneApplicationPage>
```

4-6 Arduino與行動裝置互動設計

- 實驗名稱：Arduino 控制LED亮滅實驗。
  - 建立「 Windows Phone 應用程式」，「MainPage.xaml.CS」的程式碼。
    - using System;
    - using System.Collections.Generic;
    - using System.Linq;
    - using System.Net;
    - using System.Net.Sockets;
    - using System.Windows;
    - using System.Windows.Controls;
    - using System.Windows.Documents;
    - using System.Windows.Input;
    - using System.Windows.Media;
    - using System.Windows.Media.Animation;
    - using System.Windows.Shapes;
    - using Microsoft.Phone.Controls;

4-6 Arduino與行動裝置互動設計

- 實驗名稱：Arduino 控制LED亮滅實驗。
  - 建立「 Windows Phone 應用程式」，「MainPage.xaml.CS」的程式碼。

```
namespace CS_PhoneApp1
{
    public partial class MainPage : PhoneApplicationPage
    {
        // 建構函式
        public MainPage()
        {
            InitializeComponent();
        }
```

4-6 Arduino與行動裝置互動設計

- 實驗名稱：Arduino 控制LED亮滅實驗。
  - 建立「 Windows Phone 應用程式」，「MainPage.xaml.CS」的程式碼。

```
//點亮LED按鈕事件。
    private void btn_lightsUp_Click(object sender, RoutedEventArgs e)
    {
      if (!string.IsNullOrEmpty(txt_ipAddress.Text) && !string.IsNullOrEmpty(txt_serverPort.Text))
      {
        string ipAddress = txt_ipAddress.Text.Trim();
        int serverPort = int.Parse(txt_serverPort.Text.Trim());
        Socket socket = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);
        DnsEndPoint dnsEndPoint = new DnsEndPoint(ipAddress, serverPort);
        SocketAsyncEventArgs socketAsyncEventArgs = new SocketAsyncEventArgs();
        byte[] buffer = System.Text.Encoding.UTF8.GetBytes("GET /?1 HTTP/1.1\r\nHost: " + ipAddress + "\r\nConnection:
Close\r\n\r\n");
        socketAsyncEventArgs.SetBuffer(buffer, 0, buffer.Length);
        socketAsyncEventArgs.RemoteEndPoint = dnsEndPoint;
        socketAsyncEventArgs.UserToken = socket;
        try  {
          if (socket.ConnectAsync(socketAsyncEventArgs))
          {
            txt_infomation.Text = "點亮LED指令送出成功。";
          }
        }
        catch (SocketException ex)
        {
          txt_infomation.Text = string.Format("擲出例外情況! 錯誤代碼：{0}", ex.ErrorCode);
        }
      }
    }
```

4-6 Arduino與行動裝置互動設計

- 實驗名稱：Arduino 控制LED亮滅實驗。
  - 建立「 Windows Phone 應用程式」，「MainPage.xaml.CS」的程式碼。

```
//熄滅LED按鈕事件。
    private void btn_lightsOff_Click(object sender, RoutedEventArgs e)
    {
      if (!string.IsNullOrEmpty(txt_ipAddress.Text) && !string.IsNullOrEmpty(txt_serverPort.Text))
      {
        string ipAddress = txt_ipAddress.Text.Trim();
        int serverPort = int.Parse(txt_serverPort.Text.Trim());
        Socket socket = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);
        DnsEndPoint dnsEndPoint = new DnsEndPoint(ipAddress, serverPort);
        SocketAsyncEventArgs socketAsyncEventArgs = new SocketAsyncEventArgs();
        byte[] buffer = System.Text.Encoding.UTF8.GetBytes("GET /?0 HTTP/1.1\r\nHost: " + ipAddress + "\r\nConnection:
Close\r\n\r\n");
        socketAsyncEventArgs.SetBuffer(buffer, 0, buffer.Length);
        socketAsyncEventArgs.RemoteEndPoint = dnsEndPoint;
        socketAsyncEventArgs.UserToken = socket;
        try    {
          if (socket.ConnectAsync(socketAsyncEventArgs))
          {
            txt_infomation.Text = "熄滅LED指令送出成功。";
          }
        }
        catch (SocketException ex)     {
          txt_infomation.Text = string.Format("擲出例外情況! 錯誤代碼：{0}", ex.ErrorCode);
        }
      }
    }
  }
}
```

4-6 Arduino與行動裝置互動設計

- 實驗名稱：Arduino 控制LED亮滅實驗。
  - 建立「Arduino」的程式碼：
    - //使用Arduino Ethernet Shield網卡時要引用下列兩個.h檔。
    - #include <SPI.h>
    - #include <Ethernet.h>
    - //定義第9腳位為LEDPIN。
    - #define LEDPIN 9
    - //用途：當抓取指令第一個字$時, incoming會為1(true), 表示指令的第一個字正確。
    - boolean incoming = 0;
    - //定義LED的狀態為低電位。
    - int LEDPINSTATE = 0;

4-6 Arduino與行動裝置互動設計

- 實驗名稱：Arduino 控制LED亮滅實驗。
  - 建立「Arduino」的程式碼：
    - //必須輸入你自己的Arduino Ethernet Shield網卡所設定的MAC ADDRESS。
    - //MAC ADDRESS通常可以在網卡背面找到。
    - byte mac[] = {
    - 0xAA, 0xBB, 0xCC, 0xDD, 0xEE, 0xFF };
    - //必須輸入你自己的Arduino Ethernet Shield網卡所分配的IP位址。
    - IPAddress ip(192 ,168 ,1 ,199);
    - //初始化乙太網路伺服器。
    - //一般都使用80 Port來作為網頁伺服器的通訊埠。
    - EthernetServer server(80);

4-6 Arduino與行動裝置互動設計

- 實驗名稱：Arduino 控制LED亮滅實驗。
  - 建立「Arduino」的程式碼：

```
//Arduino程式的初始設定, 僅於上電時執行一次。
void setup()
{
  //指定LEDPIN(第9腳位)為輸出。
  pinMode(LEDPIN, OUTPUT);

  //啟動乙太網路伺服器以及連線
  Ethernet.begin(mac, ip);
  server.begin();

  //設定傳輸率為9600。
  Serial.begin(9600);
  //送出序列埠訊息顯示目前的網卡IP作為提示。
  Serial.print("server is at ");
  Serial.println(Ethernet.localIP());
}
```

4-6 Arduino與行動裝置互動設計

- 實驗名稱：Arduino 控制LED亮滅實驗。
  - 建立「Arduino」的程式碼：

```
//Arduino程式的迴圈區段, 一直到斷電才結束動作。
void loop()
{
 //開始監聽連入的客戶端
 EthernetClient client = server.available();
 if (client) {
  Serial.println("new client");
  //當有客戶端連入時就繼續執行程式。
  while (client.connected()) {
   //當客戶端有送出訊息時則準備接收字元。
   if (client.available()) {
    //開始接收字元。
    char c = client.read();

    //開始讀取URL, 分析第一個字元是否為問號。
    if(incoming && c == ' '){
     incoming = 0;
    }
    //當第一個字元是問號的時候, 把incoming的狀態改為true。
    if(c == '?'){
     incoming = 1;
    }
```

4-6 Arduino與行動裝置互動設計

- 實驗名稱：Arduino 控制LED亮滅實驗。
  - 建立「Arduino」的程式碼：

```
//檢查URL中是否為指令, ?1 或者是 ?0。
      if(incoming == 1){
       Serial.println(c);

       //當URL為?1的時候。
       if(c == '1'){
        //印出ON, 可以在序列埠監控視窗中看到ON。
        Serial.println("ON");
        //將LED的狀態設定為高電位。
        LEDPINSTATE = 1;
       }
       //當URL為?0的時候。
       if(c == '0'){
        //印出OFF, 可以在序列埠監控視窗中看到OFF。
        Serial.println("OFF");
        //將LED的狀態設定為低電位。
        LEDPINSTATE = 0;
       }
      }
```

4-6 Arduino與行動裝置互動設計

- 實驗名稱：Arduino 控制LED亮滅實驗。
  - 建立「Arduino」的程式碼：

```
//當發現換行字元則可以控制LED燈的狀態並且印出結果於網頁上。
      if (c == '\n'){
        //根據LEDPINSTATE的狀態來決定LED燈的亮滅。
        digitalWrite(LEDPIN, LEDPINSTATE);
        //以下為印出結果並且呈現於網頁上
        client.println("HTTP/1.1 200 OK");
        //使用utf-8編碼, 避免瀏覽器出現亂碼的情形。
        client.println("Content-Type: text/html; charset=utf-8");
        client.println("Connnection: close");
        client.println();
        client.println("<!DOCTYPE HTML>");
        client.println("<html>");
        client.println("<title>Arduino 控制LED亮滅實驗</title>");
        //使用utf-8編碼, 避免瀏覽器出現亂碼的情形。
        client.println("<meta http-equiv=\"Content-Type\" content=\"text/html; charset=utf-8\">");
        if (LEDPINSTATE == 1)
          client.println("<h1>點亮LED。</h1>");
        else
          client.println("<h1>熄滅LED。</h1>");
        client.println("</html>");
        break;
      }
    }
   }
   //延遲一微秒, 防止接收資料出錯。
   delay(1);
   //關閉連線。
   client.stop();
  }
}
```

4-6 Arduino與行動裝置互動設計

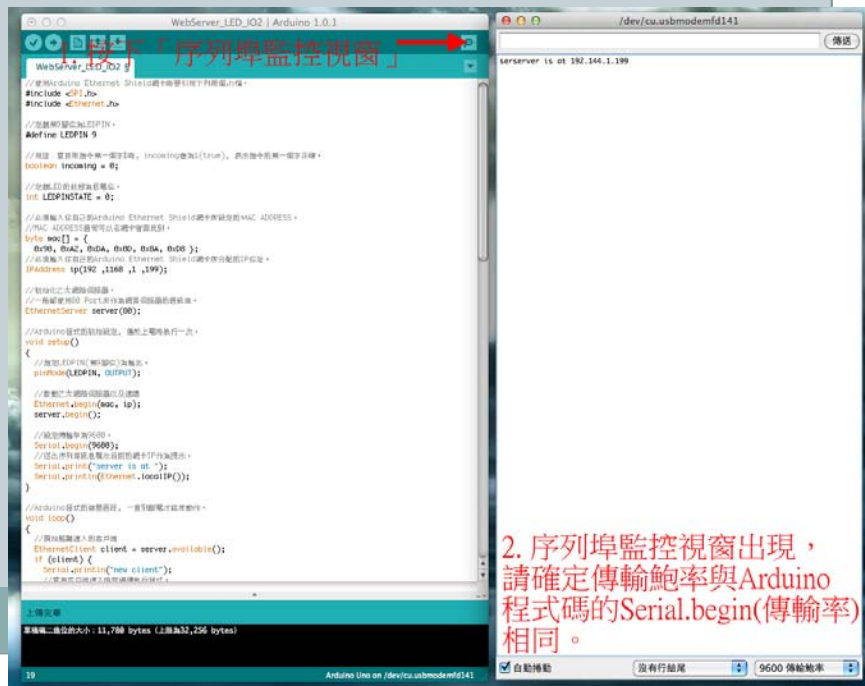
- 實驗名稱：Arduino 控制LED亮滅實驗。
  - 建立「Arduino」的接線圖：LED的長腳接到第９腳位，短腳接到第１３腳位左邊的GND腳位。

4-6 Arduino與行動裝置互動設計

- 實驗名稱：Arduino 控制LED亮滅實驗。
  - 動作說明：
    - 「Windows Phone應用程式」介面以及程式碼撰寫完成之後，請將程式「編譯」，沒有錯誤之後就可以選擇用來偵錯的裝置「Windows Phone Device、Windows Phone Emulator – 512 MB(ZH-HANT)或是Windows Phone Emulator – 256 MB(ZH-HANT) 」然後「開始偵錯」。
      - 建議使用Windows Phone Emulator – 256 MB(ZH-HANT) 即可。

4-6 Arduino與行動裝置互動設計

- 實驗名稱：Arduino 控制LED亮滅實驗。
  - 動作說明：在完成接線以及程式撰寫，我們上傳程式至Arduino UNO無誤之後，按下Arduino IDE的「序列埠監控視窗」，可以看到目前紅外線感測器元件的狀態。
  - 另外，請確定傳輸鮑率與Arduino程式碼的Serial.begin(傳輸率)相同。



4-6 Arduino與行動裝置互動設計

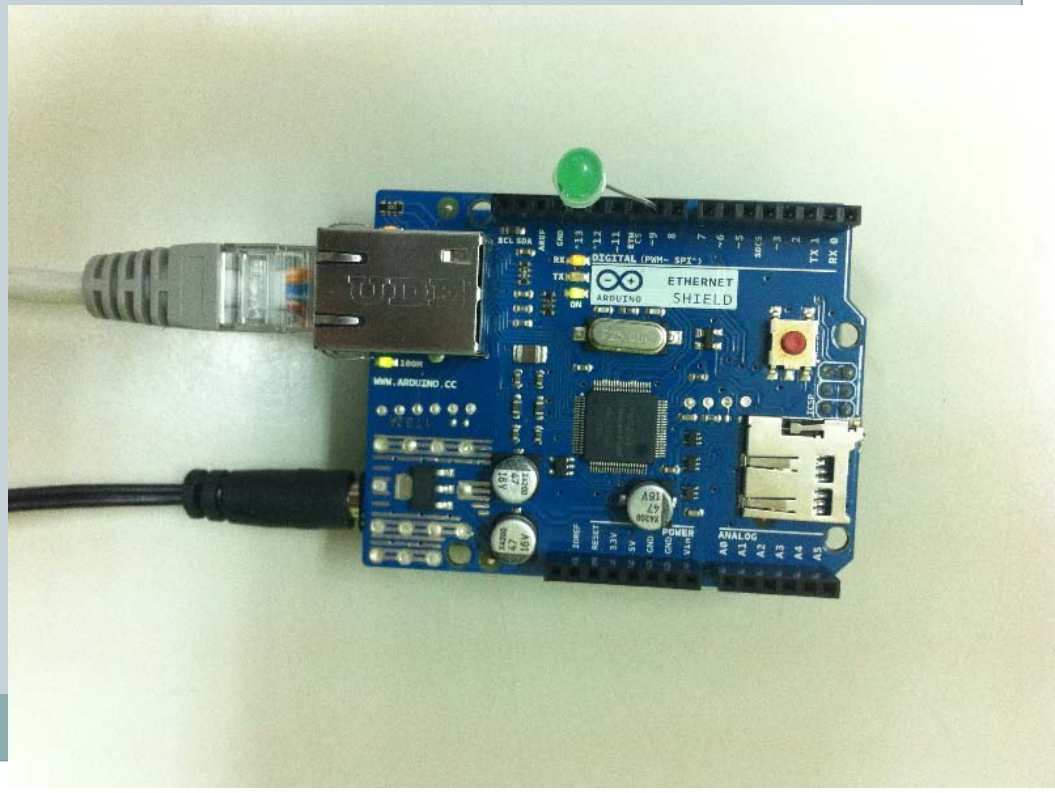
- 實驗名稱：Arduino 控制LED亮滅實驗。
  - 動作說明：
    - Windows Phone Emulator執行後會載入我們所撰寫的應用程式，可以看到主畫面如下圖，接下來就輸入「分配給Arduino Ethernet Shield網卡的IP位址」以及「 Arduino程式碼所指定的網頁伺服器Port」。

4-6 Arduino與行動裝置互動設計

- 實驗名稱：Arduino 控制LED亮滅實驗。
  - 動作說明：
    - 指定完這兩項資訊之後我們可以按下「點亮LED」然後觀察Arduino上的LED是否被點亮了。

4-6 Arduino與行動裝置互動設計

- 實驗名稱：Arduino 控制LED亮滅實驗。
  - 動作說明：
    - 接著可以按下「熄滅LED」然後觀察Arduino上的LED是否熄滅了。



4-6 Arduino與行動裝置互動設計

- 實驗名稱：Arduino 控制LED亮滅實驗。
  - 動作說明：
    - 當Arduino上的LED燈被點亮的時候，我們可以發現Arduino IDE的「序列埠監控視窗」這樣的出現訊息：
      - new client
      - ?
      - 1
      - ON
    - 而當Arduino上的LED燈被熄滅的時候，我們可以發現Arduino IDE的「序列埠監控視窗」這樣的出現訊息：
      - new client
      - ?
      - 0
      - OFF

4-6 Arduino與行動裝置互動設計

- 實驗名稱：Arduino 控制LED亮滅實驗。
  - 動作說明：
    - 由於這是藉由發送一個HTTP GET訊息給Arduino，所以我們也可以藉由一般的網頁瀏覽器來控制Arduino的LED亮滅。
    - 在網頁瀏覽器上輸入網址：
    - 點亮LED的指令為：
      http://Arduino的IP位址/?1
    - 熄滅LED的指令為：
      http://Arduino的IP位址/?0

4-6 Arduino與行動裝置互動設計

- 實驗名稱：Arduino 控制LED亮滅實驗。
  - 實驗結果：
    - 藉由這個實驗我們學習到如何使用Arduino搭配網卡擴展板達到上網傳輸資料的目的。
    - 我們還可以學習到動手撰寫Windows Phone應用程式。
  - 延伸想法：
    - 利用Arduino以及網卡擴展板來將感測器蒐集到的資訊以網頁的方式呈現。
    - 利用Arduino以及網卡擴展板來將感測器蒐集到的資訊傳送到資料庫中，然後交給動態網頁來呈現更複雜的網頁資訊。
    - 練習將自己撰寫的Windows Phone應用程式編譯成.XAP檔案後，透過Windows Phone SDK中的Application Deployment，將Windows Phone應用程式部署到Windows Phone中，以實機操作你的Windows Phone應用程式。

4-6 Arduino與行動裝置互動設計

- 實驗名稱：Arduino 控制LED亮滅實驗。
  - 將Windows Phone應用程式部署到Windows Phone中實機測試：